Naval Research Laboratory

Washington, DC 20375-5000

NRL Report 9293

One-Dimensional Translation Measurement of Speckle from Rough Rotating Objects in Ultraviolet Illumination

BONNIE LIGHT AND GARY L. TRUSTY

Applied Optics Branch
Optical Sciences Division

December 31, 1990

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 31, 1990	Interim

4. TITLE AND SUBTITLE

One-Dimensional Translation Measurement of Speckle from Rough Rotating Objects in Ultraviolet Illumination

5. FUNDING NUMBERS

PE - 64270N
WU - DN157-149

6. AUTHOR(S)

Light, Bonnie and Trusty, Gary L.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Washington, DC 20375-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL Report 9293

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Naval Air Systems Command
Washington, DC 20361-8030

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The experimental measurement of translation in one dimension from near-field speckle produced by rough rotating objects is considered in this report. Rotation rates about a single fixed axis are estimated from measured speckle intensity pattern translation values. The feasibility of translation measurements made by cross-correlation technique in a single dimension is demonstrated. A technique for image enhancement is also demonstrated. The speckle pattern images can be enhanced so that intensity distributions meet near-ideal distributions and contrast values. Speckle is produced experimentally by small (2.5-, 4.0-, 5.0-mm diameter) spherical objects with aluminum paint surfaces in coherent ultraviolet illumination. The measurements are made from imagery taken before and after object angular displacement, rather than from imagery taken of objects revolving in real time. Signal-to-noise ratios are compared for the direct evaluation of the cross-correlation function and for the evaluation of the normalized form of the cross-correlation function as well as for correlation functions for unenhanced and enhanced imagery.

14. SUBJECT TERMS

Laser speckle
Cross-correlation function
Image enhancement
Image contrast

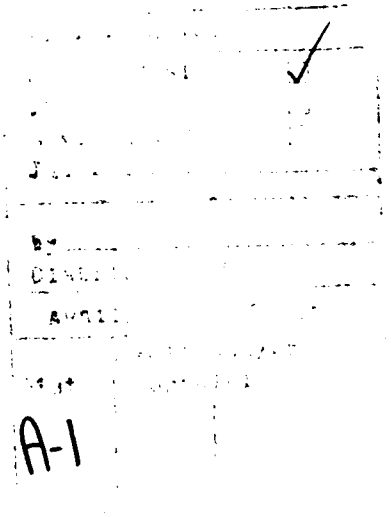
15. NUMBER OF PAGES

68

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

CONTENTS



ONE-DIMENSIONAL TRANSLATION MEASUREMENT OF SPECKLE FROM ROUGH ROTATING OBJECTS IN ULTRAVIOLET ILLUMINATION

1. INTRODUCTION

One easily observed property of spatially coherent light is the granular appearance of its reflection from a diffuse surface. At any point in space, the reflected field is the superposition of many contributing wavelets scattered from different scattering cells on the illuminated reflecting surface. The path lengths traveled by rays scattered from a surface rough on the scale of an optical wavelength differ by several or many wavelengths. The scattered wavefronts have a constant relative phase determined by the optical path length from each scatterer to the observation point; their sum produces a stationary interference pattern. Interference of the dephased but coherent wavelets results in the granular pattern of intensity that is known as speckle. Laser speckle is observed in free-space propagation as well as in imaging systems.

Quantitatively, the average size of a speckle is approximated by the relation $s = \lambda R/d$, where λ is the illuminating wavelength, R is the range from the scattering surface to the observation point, and d is the diameter of the illuminated area on the scattering surface.

Often speckle patterns are labeled as practical nuisances in coherently illuminated systems. However, monitoring a speckle pattern can produce useful information about the dynamics and surface structure of the reflecting object. This investigation uses the speckle pattern from a rotating object to monitor object angular displacement. It is observed that when an object rotates about a fixed axis, the speckle from that object translates in the direction perpendicular to the fixed axis of rotation. The displacement of the speckle intensity pattern is proportional to the angular displacement of the rotating object. The cross-correlation function of two shifted intensity patterns gives an estimate of the distance translated by the speckle pattern.

Several authors have treated this technique theoretically and experimentally. N. George [1] presented a theory for the characteristics of speckle in the far field caused by electromagnetic waves scattered by a rotating, rough surface. He derived an expression for the cross-correlation function of the electric field scattered to two different observation points.

Hayashi and Kitagawa [2] developed a method for measuring the rotation angle of a cylinder based on displacement of near-field speckle caused by cylinder surface rotation. They found that speckle translation is equal to object rotation multiplied by a constant. They also provided experimental results of their investigation.

Marron and Schroeder [3] have developed a modeling technique that treats the optical field as a sum of contributions from discrete scatterers on the surface of the object. By tracing a series of rays that coarsely sample the object, they compute relative phase shifts resulting from object rotation and calculate the speckle correlation function. They present this model as well as experimental results for objects of various surface coatings. Their work was done in visible illumination, and they recorded two-dimensional (2-D) speckle patterns for the computation of correlation functions.

Manuscript approved June 5, 1990.

Fujii and Asakura [4] have investigated the effect of surface roughness on the statistical distribution of image speckle intensity. They describe the statistical variations of speckle and how they alter speckle contrast.

This investigation is devoted to the experimental determination of object rotation inferred from the measurement of speckle translation in one direction in the near field. The speckle is generated with coherent ultraviolet radiation, and the measurement of speckle translation requires sampling the intensity patterns before and after object angular displacement. The one-dimensional (1-D) image samples are used to compute the cross-correlation function.

This investigation demonstrates the feasibility of limiting the speckle pattern sampling to the single dimension parallel to the direction of translation. By sampling in this direction, only rotations about the axis perpendicular to the sampling direction can be detected. Efforts toward real-time computational capability motivate 1-D sampling, since computations on 2-D images can be time consuming.

This investigation also explores the use of an image enhancement technique to enhance the experimentally recorded speckle patterns so that they possess near ideal statistics.

Chapter 2 of this report is a mathematical derivation of the normalized cross-correlation function of two speckle patterns. This derivation is based on the assumption of ideal speckle image intensity distributions. The theory of an image enhancement technique is also presented. The technique maps the gray levels of an image with less than ideal statistics to gray levels with a more desirable probability density function.

Chapter 3 describes the experimental demonstration of this technique, the optical system, and the computational methods used in the processing code. Exemplary results of object rotation measurement are also presented.

Chapter 4 gives a summary of the work presented and the concluding remarks.

Chapter 5 lists references cited.

Chapter 6 is an appendix that includes the source code for the programs written for this experiment.

2. THEORY

The speckle from a rotating object translates as the object rotates and is analogous to the reflection of a ray from a rotating mirror. Consider the plane reflecting mirror of Fig. 1 that is rotated through a small angle $\Delta\theta$ about an axis through its surface. When the incident ray is along the normal of the reflecting surface, the reflection is also observed along the normal. After rotation by $\Delta\theta$, the reflected ray is observed at an angle $2\Delta\theta$ from the normal. When observed at a plane a distance R from the surface of the object, the ray has translated in the observation plane a distance approximated by $2\Delta\theta R$ [5]. To estimate the angle of mirror rotation, we need only measure the translated distance of the reflected ray in the observation plane and know the range R from the scattering surface to the observation point.

Two signals representing speckle intensity vs spatial position in one dimension, $I_1(x)$ and $I_2(x)$, can be compared and examined for similarities by computing their cross-correlation function. If one is translated in position with respect to the other, the computation of the cross-correlation function shows

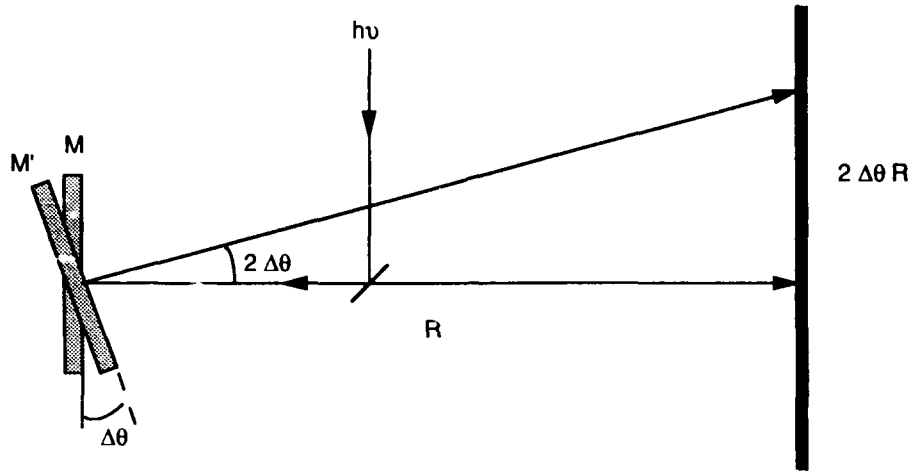


Fig. 1 — Reflected ray from rotated mirror. As mirror M rotates by $\Delta\theta$ to position M' the reflected ray is translated a distance $2\Delta\theta R$ in the observation plane.

that this function peaks at the position equivalent to the position difference of the two signals. When one pattern is a shifted version of the other, the similarities between the two become apparent only as one of the signals is shifted in position, so we correlate $I_2(x + n)$ with $I_1(n)$ over incremental values for x. The cross-correlation of two 1-D signals can be defined as a position shifted set of products, given by

$$R_I(x_1;\, x_2) = \sum_{n=0}^{N-1} I_1(n)I_2(x + n) = I_1(x)\cdot I_2(x) = \langle\, I_1(x)I_2(x)\,\rangle \tag{1}$$

for ($x = 0, 1, \ldots N$-1), where N is the number of intensity samples.

This computation can be evaluated directly in the spatial domain or by Fourier technique in the frequency domain. The application of the Fourier technique is a result of the correlation theorem. The correlation theorem states that the cross correlation of two signals corresponds to the multiplication of the Fourier transform of one signal by the complex conjugate of the Fourier transform of the other.

$$R_I(x_1;\, x_2) = \mathcal{F}^{-1}\,\{\mathcal{F}\,[I_1(x)]\cdot\mathcal{F}^{*}\,[I_2(x)]\} \tag{2}$$

where the symbol $\mathcal{F}$ indicates the Fourier transform, and $\mathcal{F}^{-1}$ indicates the inverse Fourier transform. The implementation of the direct evaluation of Eq. (1) requires a number proportional to N^2 of multiplications and additions, whereas the Fast Fourier Transform (FFT) reduces the number of computations to a value proportional to ($N \log_2 N$). Therefore, it is faster (for large N) to obtain $R_I(x_1;\, x_2)$ by employing the FFT algorithm.

In practice, sampled images of speckle are not free of noise; two successive frames are not perfectly shifted replicas, and noise is introduced through the finite number of samples taken. The unnormalized cross-correlation function of two signals generally has poor signal-to-noise ratio (SNR). Noise reduction can be achieved by normalizing the cross-correlation function. In the following discussion the development of first- and second-order speckle statistics leads to an expression for the normalized correlation coefficient, a convenient form of the normalized correlation function.

First-order statistics describe the statistical properties at a single point in space, and second-order statistics describe properties at two spatial locations. The cross-correlation function is a second-order statistical quantity because it compares two points for their similarities. To derive the first-order statistics

of speckle, analysis of a random walk in the complex plane yields statistical relations of the complex amplitude of the electric field, the intensity, and the phase of the speckle. This discussion follows that of Goodman [6].

It is necessary to put some restrictions on the manner in which the speckle is generated. Only the speckle from perfectly monochromatic light with planar wavefronts is considered, and the speckle is assumed to be polarized, i.e., the only component of the electric field is in a single direction. Although the fields are three dimensional (3-D) in space, this discussion considers only the 1-D description of the field that is described as position x, since image sampling only involves one dimension.

Let $u(x; t)$ represent the electric field at the observation point x, and at time t.

$$u(x;t) = A(x) \exp [i2\pi\nu t], \tag{3}$$

where ν is the optical frequency and A, the phasor amplitude of the field, is a complex function of position,

$$A(x;t) = |A(x;t)| \exp [i\phi(x)] \tag{4}$$

The speckle intensity is given as the square of the field, and can be represented as

$$I(x) = \lim_{T\to\infty} \int_{-\frac{T}{2}}^{\frac{T}{2}} |u(x;t)|^2 \, dt = |A(x)|^2. \tag{5}$$

The amplitude of the electric field at any one observation point can be considered as a sum of contributions from different scattering cells on the surface of the object,

$$A(x) = \sum_{k=1}^{N} \frac{1}{\sqrt{N}} a_k(x) = \frac{1}{\sqrt{N}} \sum_{k=1}^{N} |a_k| e^{i\phi_k}, \tag{6}$$

where N is the number of individual scattering cells that are inluded in the illuminated surface. The phasor amplitude can be represented by a sum of elementary phasor contributions where $|a_k|$ and ϕ_k are the amplitude and phase of the contribution from the kth scattering cell, respectively. The value of $|a_k|$ is determined by the scattering cross section of the particular cell and by the strength and uniformity of the illuminating field. The phase ϕ_k is given by the optical path length of the light as it travels from the source to the scattering cell to the observation point.

The complex addition of elementary phasors resulting in the phasor A describes a random walk. Two statisical properties of the contributing phasors are determined to be:

a. The amplitude $|a_k|$ and the phase ϕ_k of the kth elementary phasor are statistically independent of each other and of the amplitude and phases of all other elementary phasors (elementary scattering cells are unrelated, and the strength of a given scattered component is independent of its phase).

b. The phases ϕ are uniformly distributed on the primary interval $(-\pi, \pi)$ (i.e., the surface is rough compared to a wavelength, so that variations in path length result in phase differences of many times 2π radians producing a uniform distribution on the primary interval).

The real and imaginary parts of the complex amplitude are defined as

$$A^{(r)} = Re\{A\} = \frac{1}{\sqrt{N}} \sum_{k=1}^{N} |a_k| \cos\phi_k \tag{7a}$$

$$A^{(i)} = Im\{A\} = \frac{1}{\sqrt{N}} \sum_{k=1}^{N} |a_k| \sin\phi_k \, . \tag{7b}$$

The expectation values of $A^{(r)}$ and $A^{(i)}$ are

$$\langle A^{(r)} \rangle = \frac{1}{N} \sum_{k=1}^{N} \langle |a_k| \cos \phi_k \rangle = \frac{1}{N} \sum_{k=1}^{N} \langle |a_k| \mathrm{X} \cos \phi_k \rangle = 0 \tag{8a}$$

$$\langle A^{(i)} \rangle = \frac{1}{N} \sum_{k=1}^{N} \langle |a_k| \sin \phi_k \rangle = \frac{1}{N} \sum_{k=1}^{N} \langle |a_k| \mathrm{X} \sin \phi_k \rangle = 0 \, , \tag{8b}$$

where property (a) is used to average over $|a_k|$ and ϕ_k separately and property (b) yields that $\langle \cos \phi_k \rangle = \langle \sin \phi_k \rangle = 0$. The expectation values of the squared phasor real and imaginary parts become

$$\langle [A^{(r)}]^2 \rangle = \frac{1}{N} \sum_{k=1}^{N} \sum_{m=1}^{N} \langle |a_k||a_m| \rangle \langle \cos \phi_k \cos \phi_m \rangle = \frac{1}{N} \sum_{k=1}^{N} \frac{\langle |a_k|^2 \rangle}{2} \tag{9a}$$

$$\langle [A^{(i)}]^2 \rangle = \frac{1}{N} \sum_{k=1}^{N} \sum_{m=1}^{N} \langle |a_k||a_m| \rangle \langle \sin \phi_k \sin \phi_m \rangle = \frac{1}{N} \sum_{k=1}^{N} \frac{\langle |a_k|^2 \rangle}{2} \, , \tag{9b}$$

from the fact that

$$\langle \cos \phi_k \cos \phi_m \rangle = \langle \sin \phi_k \sin \phi_m \rangle = \begin{array}{ll} 1/2 & k=m \\ 0 & k \neq m \end{array} \, . \tag{10}$$

The product becomes

$$\langle A^{(r)} A^{(i)} \rangle = \frac{1}{N} \sum_{k=1}^{N} \sum_{m=1}^{N} \langle |a_k||a_m| \mathrm{X} \cos \phi_k \sin \phi_m \rangle = 0 \, , \tag{11}$$

from the fact that $\langle \cos \phi_k \sin \phi_m \rangle = 0$.

The real and imaginary parts of the complex field have zero means, identical variances, and are uncorrelated. The central-limit theorem states that, for statistically independent random variables, *the probability density of the sample mean tends to become Gaussian as the number of statistically independent samples is increased without limit*, regardless of the probability density of the random

variable or process being sampled as long as it has a finite mean and a finite variance [7]. If the number of scattering cells is allowed to get large, it follows from the central limit theorem that as $N \to \infty$ $A^{(r)}$ and $A^{(i)}$ are asymptotically Gaussian. This and Eqs. (8) and (9) yield that the joint probaility density function of the real and imaginary parts of the field asymptotically approach

$$p_{r,i}(A^{(r)},A^{(i)}) = \frac{1}{2\pi\sigma^2} \exp \frac{-([A^{(r)}]^2+[A^{(i)}]^2)}{2\sigma^2}, \tag{12}$$

where

$$\sigma^2 = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \frac{\langle |a_k|^2 \rangle}{2}. \tag{13}$$

Such a density function is known as a circular Gaussian density function, since contours of constant probability density are circles in the complex plane. From this the phasor amplitude A is referred to as a circular and complex Gaussian random variable.

With knowledge of the statistics of the complex amplitude, the corresponding statistical properties of the intensity of a speckle pattern can be computed. The intensity I and phase ϕ are given in Eqs. (4) and (5) and can be expressed in terms of the real and imaginary parts of the complex field amplitude

$$\begin{aligned} A^{(r)} &= \sqrt{I} \cos \phi \\ A^{(i)} &= \sqrt{I} \sin \phi. \end{aligned} \tag{14}$$

To find the joint probability density function of I and ϕ, the technique for transformations of random variables given by Davenport and Root [7] can be applied

$$p_2(y_1,\ldots,y_N) = p_1(x_1 = f_1,\ldots,x_N = f_N) \parallel J \parallel . \tag{15}$$

The joint density function is expressed in terms of the joint density function for $A^{(r)}$ and $A^{(i)}$ as

$$p_{I,\phi}(I,\phi) = p_{r,i}(A^{(r)}, A^{(i)}) \parallel J \parallel = p_{r,i}(\sqrt{I} \cos \phi, \sqrt{I} \sin \phi) \parallel J \parallel , \tag{16}$$

where $\|J\|$ is the Jacobian of the transformation

$$|| J || = \left\| \begin{array}{cc} \frac{\partial A^{(r)}}{\partial I} & \frac{\partial A^{(r)}}{\partial \phi} \\ \\ \frac{\partial A^{(i)}}{\partial I} & \frac{\partial A^{(i)}}{\partial \phi} \end{array} \right\|, \tag{17}$$

and $\| \ldots \|$ symbolizes the modulus of the determinant. Substituting Eq. (12) into Eq. (16), yields

$$p_{I,\phi}(I,\phi) = \frac{1}{4\pi\sigma^2} \exp\left(-\frac{I}{2\sigma^2}\right) \qquad \textit{for } I \geq 0,\ -\pi \leq \phi < \pi$$
$$0 \qquad \textit{otherwise.} \tag{18}$$

The marginal-probability density function of the intensity alone can be found from the joint-probability density function by integrating it over all possible values of the undesired variable, giving the marginal probability density of the intensity to be

$$p_I(I) = \int_{-\pi}^{\pi} p_{I,\phi}(I,\phi)d\phi = \frac{1}{2\sigma^2} \exp\left(-\frac{I}{2\sigma^2}\right) \qquad \textit{for } I \geq 0$$
$$0 \qquad \textit{otherwise.} \tag{19}$$

Similarly, the marginal density function of the phase is given by

$$p_\phi(\phi) = \int_{-\infty}^{\infty} p_{I,\phi}(I,\phi)dI = \frac{1}{2\pi} \qquad -\pi \leq \phi < \pi$$
$$0 \qquad \textit{otherwise.} \tag{20}$$

It can be concluded that the intensity observed in a polarized speckle pattern obeys negative exponential statistics, while the phase obeys uniform statistics.

The n^{th} moment $\langle I^n \rangle$ of intensity is

$$\langle I^n \rangle = n!(2\sigma^2)^n = n! \langle I \rangle^n , \tag{21}$$

so, the second moment is given by

$$\langle I^2 \rangle = 2(2\sigma^2)^2 = 2\langle I \rangle^2$$
$$\langle I^2 \rangle = 2\langle I \rangle^2 \tag{22}$$

from the definition of variance,

$$\sigma_I^2 = \langle I^2 \rangle - \langle I \rangle^2 = \langle I \rangle^2 . \tag{23}$$

A measure of the contrast of a speckle pattern is the ratio $C = \sigma_I / \langle I \rangle$. We observe that the contrast of a polarized pattern is unity, when the speckle is fully developed, that it is created with monochromatic radiation containing only a single polarization free from any stray light, and that it is composed of phasor contributions with phases distributed over the entire primary interval. For notational neatness, I_k represents $I(x_k)$ and likewise for A_k.

To develop an expression for the normalized cross-correlation function, second-order speckle statistics must be discussed. It is desired to calculate the cross corelation of the intensity distribution given in Eq. (5)

$$I_k = |A_k|^2 \tag{24}$$

by using the cross-correlation function, Eq. (1)

$$R_I(x_1; x_2) = \langle I_1 I_2 \rangle, \tag{25}$$

where the average is over an ensemble of rough surfaces. To calculate this cross-correlation function, we use the fact that, for a rough surface, the field A_k is a circular complex Gaussian random variable at each x. For such fields, the cross-correlation function of the intensity can be expressed in terms of the cross-correlation function of the fields given by

$$J_A(x_1; x_2) = \langle A_1 A_2^* \rangle, \tag{26}$$

which is referred to as the mutual intensity of the fields.

For circular-complex Gaussian fields, the required relation between R_I and J_A is

$$R_I(x_1; x_2) = \langle I_1 \rangle \langle I_2 \rangle + |J_A(x_1; x_2)|^2 , \tag{27}$$

where the fact that $I_A(x_k; x_k) = \langle I_k \rangle$ was used [8]. To calculate R_I it is now only necessary to calculate the mutual intensity J_A.

Following Goodman [9], the calculation of the mutual intensity in the observation plane begins with a fundamental relationship between the fields $\alpha(\zeta, \eta)$ at the scattering surface and the fields $A(x, y)$ at the observation point. It is necessary to discriminate between the near- and far-field speckle. When the diffraction integral can be evaluated with the far-field approximation, it is referred to as Fraunhoffer diffraction. Diffraction in the near-field case is referred to as Fresnel diffraction. The Rayleigh criterion states that when either the source of the detector are within

$$\frac{\pi a^2}{\lambda} \tag{28}$$

(where a is the aperture diameter) of the aperture, the form of the integral follows the Fresnel approximation [8]. In the present context, since the aperture measures on the order of 1 mm and the wavelength measures on the order of 300 nm the result of the expression in Eq. (28) is approximately 1/3 of 1 km. Since the detector is well within this range, it is the near-field that we desire. The desired relation is the Huygens-Fresnel principle, expressed in the Fresnel approximation as

$$A(x,y) = \frac{1}{\lambda z} \exp\left[\frac{-i\pi}{\lambda z}(x^2+y^2)\right] \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \alpha(\zeta,\eta) \cdot \exp\left[\frac{-i\pi}{\lambda z}(\zeta^2+\eta^2)\right] \cdot \exp\left[\frac{i2\pi}{\lambda z}(x\zeta+y\eta)\right] d\zeta d\eta, \tag{29}$$

where $A(x, y)$ is the field at the observation point and $\alpha(\xi,\eta)$ is the field at the scattering surface [8]. If $A(x_1, y_1)$ is expressed as such an integral over variables of integration (ζ_1, η_1) and $A(x_2, y_2)$ over variables (ζ_2, η_2), then substitution of this expression in Eq. (26) followed by an interchange of orders of integration and averaging yields the following relation between the mutual intensity J_A in the observation region and the mutual intensity J_α in the scattering plane

$$J_A(x_1,y_1;x_2,y_2) = \frac{1}{\lambda^2 z^2} \exp\left[\frac{-i\pi}{\lambda z}\left(x_1^2-x_2^2+y_1^2-y_2^2\right)\right] \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} J_\alpha(\zeta_1,\eta_1;\zeta_2,\eta_2)$$
$$\cdot \exp\left[\frac{-i\pi}{\lambda z}\left(\zeta_1^2 - \zeta_2^2 + \eta_1^2 - \eta_2^2\right)\right] \cdot \exp\left[\frac{i2\pi}{\lambda z}\left(x_1\zeta_1+y_1\eta_1-x_2\zeta_2-y_2\eta_2\right)\right] \tag{30}$$
$$d\zeta_1\, d\eta_1\, d\zeta_2\, d\eta_2.$$

This can be evaluated by making two simplifications. First, since only the modulus of J_A is of concern, the initial exponential factor in (x_1,y_1) and (x_2,y_2) is dropped. Second, it is assumed that the microstructure of the scattering surface is so fine that $J_\alpha(\zeta_1,\eta_1;\zeta_2\eta_2)$ can be approximated as

$$J_\alpha(\zeta_1,\eta_1;\zeta_2,\eta_2) \approx \kappa\, P(\zeta_1,\eta_1)P^*(\zeta_2,\eta_2)\, \delta(\zeta_1-\zeta_2,\eta_1-\eta_2), \tag{31}$$

where κ is a proportionality constant, the function $P(\zeta,\eta)$ represents the amplitude of the field incident on the scattering cell, and $\delta(\zeta,\eta)$ is a 2-D delta function. The result of these simplifications is

$$J_A(x_1,y_1;x_2,y_2) = \frac{\kappa}{\lambda^2 z^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |P(\zeta_1,\eta_1)|^2 \exp\frac{i2\pi}{\lambda z}\left[\zeta_1(x_1-x_2) + \eta_1(y_1-y_2)\right] d\zeta_1 d\eta_1 . \tag{32}$$

Thus the mutual intensity of the observed fields depends only on the difference of coordinates in the x and y direction and is given, up to multiplicative constants, by the Fourier transform of the intensity distribution $|P(\zeta,\eta)|^2$ incident on the scattering cell. Born and Wolf [10] note that this integral is analogous to the Van Cittert-Zernike theorem of classical coherence theory.

The normalized version of the mutual intensity, known as the correlation coefficient, can be defined by Ref. [10]

$$\mu_A(x_1,y_1;x_2,y_2) = \frac{J_A(x_1,y_1;x_2,y_2)}{\left[J_A(x_1,y_1;x_1,y_1)\, J_A(x_2,y_2;x_2,y_2)\right]^{\frac{1}{2}}} . \tag{33}$$

By using Eq. (32), the normalized correlation coefficient is expressed as

$$\mu_A(x_1,y_1;x_2,y_2) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |P(\zeta,\eta)|^2 \exp\left[i\frac{2\pi}{\lambda z}\,\zeta(x_1;x_2) + \eta(y_1;y_2)\right] d\zeta d\eta}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |P(\zeta,\eta|^2 d\zeta\, d\eta} . \tag{34}$$

From Eqs. (27) and (33) and $J_A(x_k,y_k; x_k,y_k) = \langle I_k \rangle$ the cross correlation of the speckle intensity is expressed as

$$R_I(x_1, y_1; x_2, y_2) = \langle I_1 \rangle \langle I_2 \rangle \left[1 + |\mu_A(x_1,y_1;x_2,y_2)|^2\right] . \tag{35}$$

Finally, $|\mu_A(x_1,y_1; x_2,y_2)|^2$ can be given as a function of the 1-D coordinate x as

$$|\mu_A(x_1; x_2)|^2 = \frac{\langle I_1 I_2 \rangle}{\langle I_1 \rangle \langle I_2 \rangle} - 1. \tag{36}$$

Marron [11] demonstrates that the inherent positive correlation between the noise of the numerator and that of the denominator causes the SNR of the ratio to be consistently larger than the SNR of the numerator alone. The estimator $\langle I_1 \rangle \langle I_2 \rangle$ applies by virtue of the negative exponential speckle statistics, which obey $\langle I^r \rangle = r! \langle I \rangle^r$.

This treatment has been given for speckle with a negative exponential intensity distribution. Under real conditions, the speckle intensity distribution varies as a function of the surface roughness. To quantify this dependence we consider the measure of contrast given by $C = \sigma_I / \langle I \rangle$. Sprague [12] reveals that the contrast of the speckle produced near the image plane is related to the surface roughness, when the roughness and coherence length of the illuminating light are approximately equal im magnitude.

Consider formation of a speckle pattern reflected from a rough surface object. Substituting the general expression for the phasor amplitude of the field as given by Eq. (6) into the expression for the intensity of the pattern, as given by Eqs. (5) and (33) the result is,

$$I(x) = \left| \frac{1}{\sqrt{N}} \sum_{k=1}^{N} |a_k| \, e^{i\Phi_k} \right|^2 . \tag{37}$$

This equation indicates that at a certain observation point the intensity is given by superposition of phase variations over the incident light waves.

Three effects have been identified as the cause of the changes in the optical field during object rotation [3]:

(a) The individual scattering cells that enter the summation of Eq. (33) change as scatterers move into and out of the illuminated region of the object. The severity of this effect depends on the extent of the illuminated region.

(b) The amplitude of a scatterer $|a_k|$ changes with angular displacement. Since $|a_k|$ is determined by the scattering cross section, angular displacement can alter the magnitude of this value. An example is a scattering cell that is faceted and gives preference to certain orientations constantly changing with object rotation.

(c) The motion of a scattering cell incurs continual changes in the path length from the cell to the observation point, thus regulating the phase of the contribution.

The conclusion reached by Marron and Schroeder [3] as well as by Leader [13] is that the effect described in subparagraph (c) above accounts for speckle translation almost exclusively in practical situations.

As phase variations over the illuminated area increase, intensity variations in the observation plane also increase and yield increased contrast in the speckle pattern. Likewise, as the surface becomes smooth, the contrast decreases and finally approaches zero contrast.

Fujii and Asakura [4] measured the probability density of the intensity as a function of surface roughness, and Fig. 2 shows their experimental results. The intensity has been normalized by the average intensity $\langle I \rangle$, and the probability has been normalized so that the total integration of the probability density function for each surface is unity. This figure shows that the smoothest surface examined (curve A) produced an intensity distribution somewhere between Poissonian and Gaussian, and the roughest surface (curve D) produced an intensity distribution that was negative exponential.

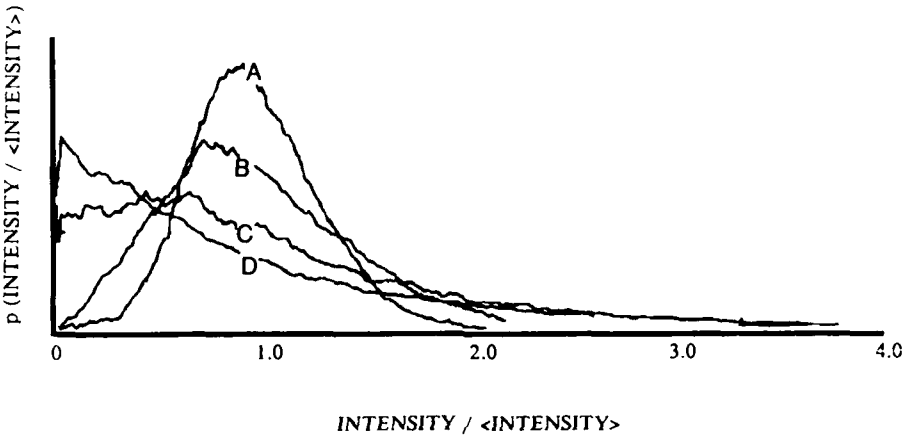


Fig. 2 — Probability density function p(INTENSITY) of speckle intensity variations for four sample objects with varying surface roughness. Curve A represents the smoothest surface examined, and curve D the roughest surface examined, with curves B, C representing surfaces of intermediate roughness. Reproduced from Fujii and Asakura [4].

The normalized correlation coefficient $|\mu_A|^2$ was derived for an intensity pattern with negative exponential probability density. Speckle, experimentally recorded, may not always display this statistical characteristic. Techniques in image enhancement can be used to conform the probability density of a recorded speckle pattern image to the appropriate distribution.

Techniques in image processing can be divided into frequency-domain methods and spatial-domain methods. Processing techniques in the first category are based on computations and manipulations of the Fourier transform of the image. The spatial domain refers to the image plane itself, and approaches are based on direct manipulation of the pixel values in the image.

When the image intensity probability density does not conform to the specified statistics, probability density enhancements techniques (processing in the spatial domain) may be used.

A histogram of the gray-level content of an image provides a description of the appearance of the image. The process of histogram enhancement involves manipulating the intensity probability density of the image by computing a gray level mapping for each gray level in the original image. The technique described is that of Gonzales and Wintz [14].

Let the variable r represent the normalized pixel intensity value in the original image. The value of r is always in the range $0 \leq r \leq 1$, where the value *0* represents black and the value *1* represents white in the gray scale. For any r in the interval (0, 1) a transformation can be defined of the form $s = T(r)$, which maps each input pixel intensity r to an output pixel intensity s. It is assumed that this transformation function satisfies the conditions,

(a) $T(r)$ is single-valued and monotonically increasing in the interval $0 \leq r \leq 1$ and

(b) $0 \leq T(r) \leq 1$ for $0 \leq r \leq 1$.

Condition (a) preserves the order from black to white in the gray scale, and condition (b) guarantees a mapping that is consistent with the allowed range of pixel intensities. The inverse transformation from s to r can be defined by $r = T^{-1}(s)$ where $0 \leq s \leq 1$.

If r and s can be treated as continuous variables, the original and transformed gray levels can be characterized by their probability density functions $p_r(r)$ and $p_s(s)$, respectively. From probability theory it follows that if $p_r(r)$ and $T(r)$ are known, assuming $T^{-1}(s)$ satisfies condition (a), then the probability density function of the transformed gray levels is given by

$$p_s(s) = \left[p_r(r) \frac{dr}{ds}\right]_{r=T^{-1}(s)} . \tag{38}$$

The first step in mapping the image gray levels to fit a specified probability density function is to equalize the histogram of the original image. Consider the function

$$s = T(r) = \int_0^r p_r(w)dw \qquad 0 \leq r \leq 1 , \tag{39}$$

where w is a dummy variable of integration. This function is recognized as the cumulative distribution function of r. The derivative of s with respect to r is given as

$$\frac{ds}{dr} = p_r(r). \tag{40}$$

Substituting Eq. (38) yields

$$p_s(s) = \left[p_r(r) \frac{1}{p_r(r)}\right]_{r=T^{-1}(s)} \tag{41}$$

$$= 1 \qquad 0 \leq s \leq 1,$$

which is a uniform density in the interval defined.

If the desired image were available, its levels could also be equalized by using the transformation function

$$v = G(z) = \int_0^z p_z(w)dw, \tag{42}$$

where $p_z(z)$ is the desired probability density function, v represents the pixel intensities of the image with equalized histogram, and $z = G^{-1}(v)$. Note that $p_s(s)$ and $p_v(v)$ would be identical uniform densities. Thus, $z = G^{-1}(s)$ results from substituting s for v in the inverse function.

Equalizing the levels of the original image by using Eq. (39), specifying the desired function and obtaining the transformation function $G(z)$ by using Eq. (42), and applying the inverse transformation function $z = G^{-1}(s)$ to the levels obtained from equalization of the original image yields an image with gray levels distributed as specified by $p_z(z)$.

3. EXPERIMENT

The demonstration of this technique was performed by using a Helium-Cadmium (He-Cd) laser operating in the ultraviolet at $\lambda = 325$ nm, objects with different surface coatings, and an image acquisition, processing, and display system. Small target sizes, short ranges, and a simple-compact optical system permitted the system to be set up on a single optical table. To avoid the difficulty of timing object rotation with image acquisition, the object was manually rotated between successive still speckle pattern captures.

Figure 3 shows the optical system schematically. The He-Cd laser outputs radiation at two wavelengths, 325 nm in the ultraviolet and 442 nm in the blue. The motivation for using ultraviolet light to illuminate the objects was twofold. Since it was known that the statistical distribution of image speckle intensity was a function of surface roughness [4], it was initially thought that surfaces of varying roughness could possibly be discriminated with short wavelength illumination. We hoped that targets with smooth surface roughness could be distinguished from targets of coarser surfaces through differences in speckle image intensity contrast values. The output of the He-Cd laser at $\lambda = 325$ nm is approximately 5 mW — not much power for the limited sensitivity of the detection system. The generation of speckle intensity patterns that were not of ideal statistics provided a good opportunity to demonstrate the application of an image enhancement technique. A blue filter was placed in front of the output coupler of the laser, thus transmitting only ultraviolet radiation into the optical system.

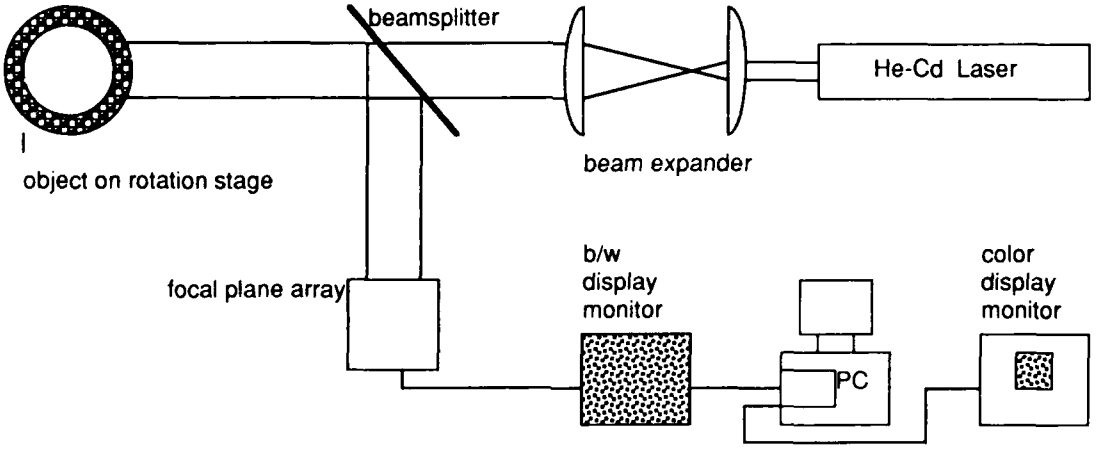


Fig. 3 — Optical system used in the experiment

Polarized ultraviolet light from the laser source passed through a pair of calcium fluoride lenses to produce a beam with planar wavefronts. The collimated beam passed through a wedged beamsplitter coated on the incident surface for antireflection at 325 nm and designed for a 45°, 50% reflection. The

light incident on the surface of the target was then scattered into all space around the target. To achieve monostatic viewing, the radiation that was scattered directly back along the direction of incidence was sampled. The light scattered from the object was reflected by the beamsplitter onto the focal plane array. It is assumed that the scattering surface of the object only depolarizes a small fraction of the incident radiation. The range R from the object rotation axis to the front surface of the focal plane was 23.60 cm.

The targets were made of small (2.5-, 4.0-, 5.0-mm diameter) spherical objects each supported by a single antivibration graphite supporting rod (mechanical pencil lead) and were mounted in the center of an Oriel precision rotation state. This arrangement insured fixed axis rotation with as much precision and control of the rotation as possible. To measure the angular rotation of the stage, a vernier scale is provided. The scale is not entirely linear throughout its range, so Oriel provides a table of scale readings and their coresponding angular displacements. The measurements during the experiment were made in the region of the scale where each integer value on the scale represented on angular displacement of 0.282 mrad. Measurements were made of angular displacements from 1 step (0.282 mrad) to 15 steps (4.230 mrad).

The objects were metallic with surfaces of machine quality smoothness. A search for surfaces that could be applied to the object and that exhibited sufficient ultraviolet reflectivity was conducted. Surfaces tested included aluminum paint, gold paint, silver paint, Newport HC 560 Retro-Paint (a silver paint with suspended glass beads) and white paint. None, except the aluminum paint, had sufficiently high reflectivity for the limited ultraviolet sensitivity of the camera. In fact the retropaint acted as a very poor reflector of ultraviolet light, a fact probably attributable to the absorption of ultraviolet light by the glass beads suspended in the paint.

The two targets that generated the most successful measurements had outermost surfaces of aluminum paint. One of the 2.5-mm targets was coated only with aluminum paint and one of the 4.0-mm spheres was coated first with Newport Retro-Paint and given a top coat of aluminum paint, thus achieving a noticeably rough surface structure. Results of measurements made with the 4.0-mm object are described in this chapter.

Speckle intensity images were recorded with a General Electric Charge Injected Device video camera, model TN2710. The camera resolution was 754 pixels in the horizontal and 484 pixels in the vertical. The camera was equipped with a quartz faceplate with a 200 to 1100 nm spectral response and had a variable gain adjustment and a clock noise filter. The pixel size was 12.0 μm in the horizontal (direction of translation) and 13.8 μm in the vertical with negligible space between pixels. To produce monostatic alignment, the central pixel was aligned with the optical axis. Images were displayed on a black and white video monitor for real-time, quick-look capability. The video was sampled and digitized by a PCVISION frame grabber made by Imaging Technology installed in a Compaq 386/25 MHz personal computer.

The imaging system sampled speckle intensity patterns by performing spatial digitization and gray level quantization. Essentially the speckle intensity pattern was sampled twice, once by the detector array and subsequently by the frame grabber. The frame grabber digitized the standard RS-170 analog signal from the camera 512 times in each row. Even though the resolution of the camera was 754 $\times$ 484 pixels, the final image resolution was limited by the 512 $\times$ 480 pixel resolution of the frame grabber. Both systems performed a horizontal raster scan on the sampled image.

To compute the distance of speckle translation on the focal plane, a conversion from frame grabber pixels to camera pixels must be computed. Each detector pixel is 12.0 μm wide and the existing 754 pixels produce an image format that is 9.048 mm wide. When the image format width is divided by the 512 pixels represented by the frame grabber, each pixel on the frame grabber represents 17.7 μm

on the focal plane surface. This information indicates that the maximum sample length (256 pixels on the frame grabber) would represent a rotation of 38.4 mrad given a range R = 23.60 cm. This is the equivalent field of view of the detector, but it is not the value of maximum measurable object rotation, because two images representing a displacement of this magnitude would not possess enough overlap signal to provide a meaningful correlation.

The image processing code was written in Fortran, compiled with the Microsoft Fortran Compiler and linked with the ImageTool [15] library of calls written for use with the frame grabber. Three menu driven executable codes were written to perform all of the image storage and processing tasks. The first program generated simulated speckle-intensity image pairs based on user specified intensity probability density, average intensity, and translation distance. The second program allowed the user to store speckle-intensity pattern images to disk as a binary file for later retrieval and performed image enhancement on images retrieved from disk. The third program digitized real-time imagery or retrieved images from disk, displayed them on the color video monitor, provided the user with statistical information about the image intensity distribution, and performed the computations of Eqs. (1) and (36). The source code for each of the programs is given in Appendix A.

The first step in processing an image pair is to capture the desired pair of images. The image processing hardware and accompanying experiment software provide for a real-time video display of the imaged speckle. The program that performs image capture gives the user a real-time display of the imagery and prompts the user to select the imagery desired. Normally, the user would set up the target, adjust its orientation so that a desirable speckle pattern is visible in the sampled area of the display, and would indicate to the computer that the desired first image is on the detector. The computer then records this image and prompts the user to manually rotate the target and indicate to the computer that the second frame is appropriately displayed on the detector. This pair of image samples is written to disk for later retrieval. All images were taken with the variable camera gain set at the normal position.

For the case of simulated speckle, to obtain a pair of speckle intensity images is a matter of generating speckle from a random number generation algorithm. The algorithms are adapted from published codes [16] for random number generation and include both generation of random numbers with negative exponential probability density function and random numbers with Poissonian probability density function. The speckles are represented by the generated random numbers normalized to the 256 gray levels. Rather than creating speckles many pixels wide, each simulated speckle is the size of a single pixel in the image. Because of this, the simulated patterns have a very different appearance than the experimentally recorded speckle patterns. They are displayed on the color monitor and recorded in the same manner as the experimentally recorded speckle patterns.

Figures 4a and 4b show reproduced examples of captured 256 × 256 pixel speckle intensity images. Figure 4a shows a simulated speckle pattern, and Fig. 4b shows speckle from the 4.0-mm diameter object coated with aluminum paint over retropaint. This speckle pattern was recorded at a range of R = 21.60 cm from the surface of the sphere. The size of the average speckle is on the order of 0.05 mm on the face of the detector.

Samples of 1-D linear pixels taken across the center row of the first and translated image are used to compute the cross-corelation function. The processing program extracts the center row of the 256 × 256 pixel image and stores the 256 pixel values in memory for processing. Figures 5(a) and 5(b) show intensity profiles of the speckle intensity of the 1-D samples taken from the simulated speckle intensity image before and after translation. The average intensity of the pixel values is 8.0 and 7.8 for Fig. 5(a) and Fig. 5(b) respectively. The corresponding contrast value for both images is 0.38. This simulated

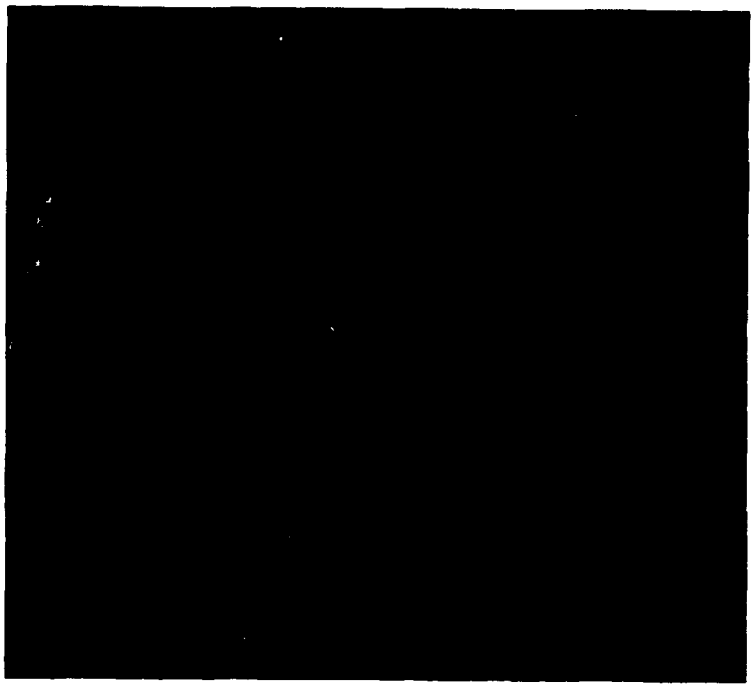


Fig. 4(a) — Simulated speckle intensity image. Reproduced from a photograph of a 256 × 256 pixel image.

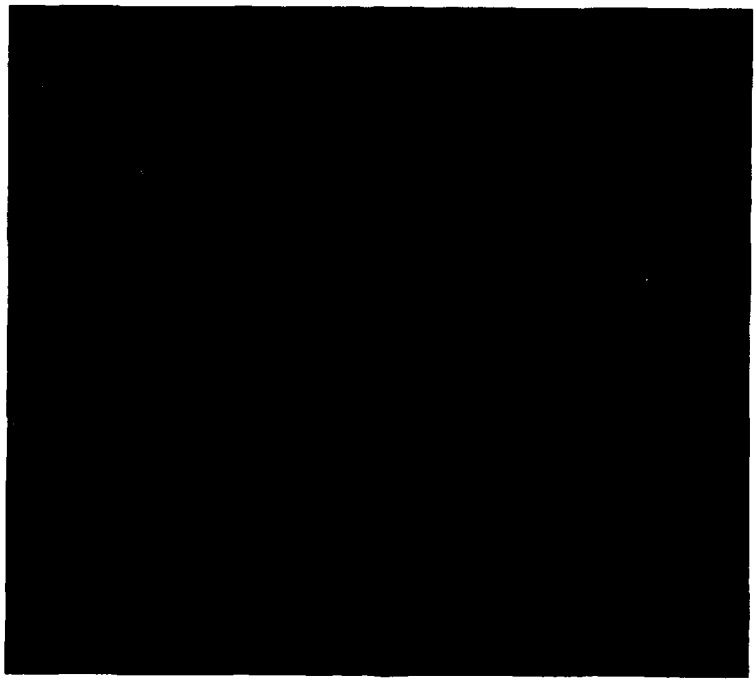


Fig. 4(b) — Experimentally recorded speckle intensity image. Reproduced from a photograph of a 256 × 256 pixel image.

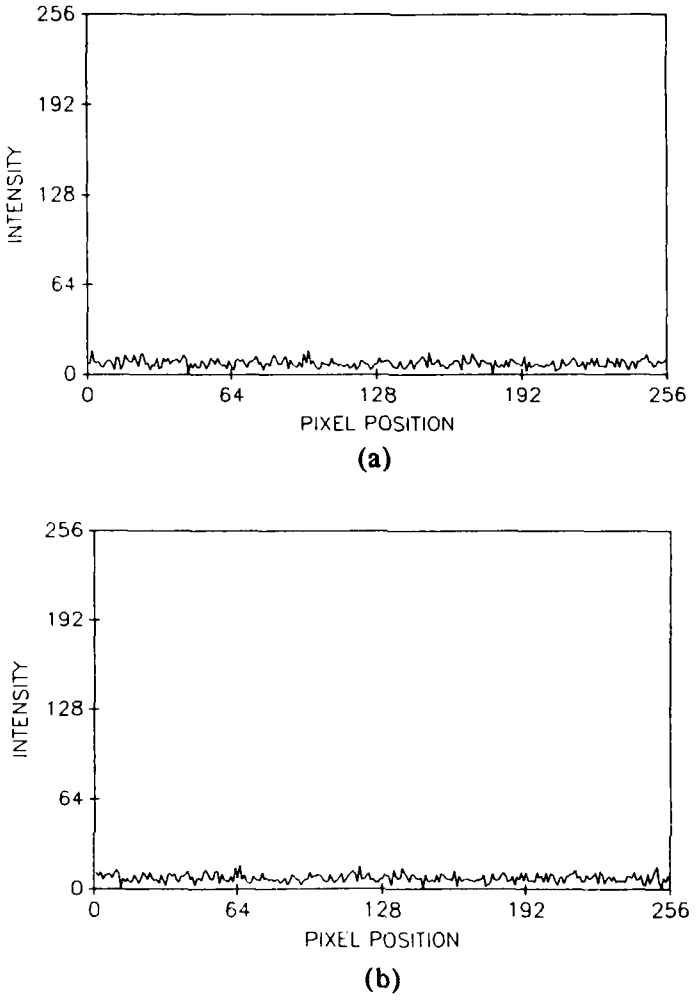


Fig. 5 — Profile of simulated speckle intensity (a) before object rotation; (b) after object rotation

speckle was generated to match the statistical properties of the experimentally collected speckle to demonstrate the image enhancement technique described.

Figures 6(a) and 6(b) show intensity profiles for the 1-D samples taken from the experimentally recorded speckle images. The average pixel intensities are 7.5 and 6.8 for these two image samples, and the contrast values are 0.46 and 0.51. Note that in both Figs. 5(a) and 5(b) and in Figs. 6(a) and 6(b) the translation of the speckle is visible between the respective figures (a) and (b). Both parts (b) of the figures are translated versions of parts (a) of the figures, with a left shift of approximately 30 pixels.

To generate the enhanced image, the theory of image enhancement discussed in Section 2 is implemented. The algorithm computes the probability and cumulative density functions of the original image and then maps the original image to new levels specified by a negative exponential probability density function, as given in Eq. (19). To specify a negative exponential probability density function, an initial value of average intensity must be given. The contrast of the new image is compared to unity, and if it is not between 0.95 and 1.0, the process repeats with an incremented value specified as the average intensity of the new image. The output is a new representation of the original image. The

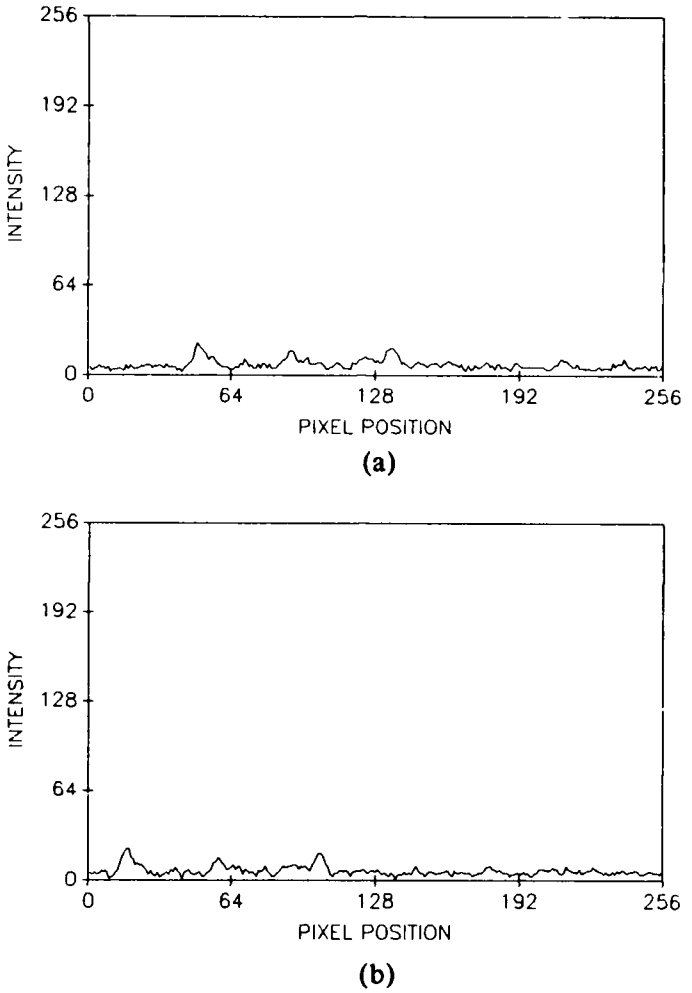


Fig. 6 — Profile of experimentally recorded speckle intensity (a) before object rotation; (b) after object rotation

increased dynamic range of the pixel intensities in the new image has a profound effect on the visible appearance of the image.

Figures 7(a) and 7(b) show the magnitude of the intensity of the enhanced simulated image samples. The average intensity value is now 56.5 for both images, and the contrast has been greatly enhanced to unity, the ideal value for fully developed speckle.

Figures 8(a) and 8(b) show the magnitude of the intensity of the enhanced speckle that was experimentally recorded. The average values for the enhanced speckle are 50.8 and 49.7, and both also have a contrast of unity.

Figures 9(a) and 9(b) indicate the probability density functions for the original and enhanced simulated speckle. The original simulated speckle of Fig. 9(a) was specified to have a Poissonian probability density function with an average value of 8.0 and a gain of unity. The enhanced simulated speckle of Fig. 9(b) has a density function that covers the entire range of pixel values, 0 to 255.

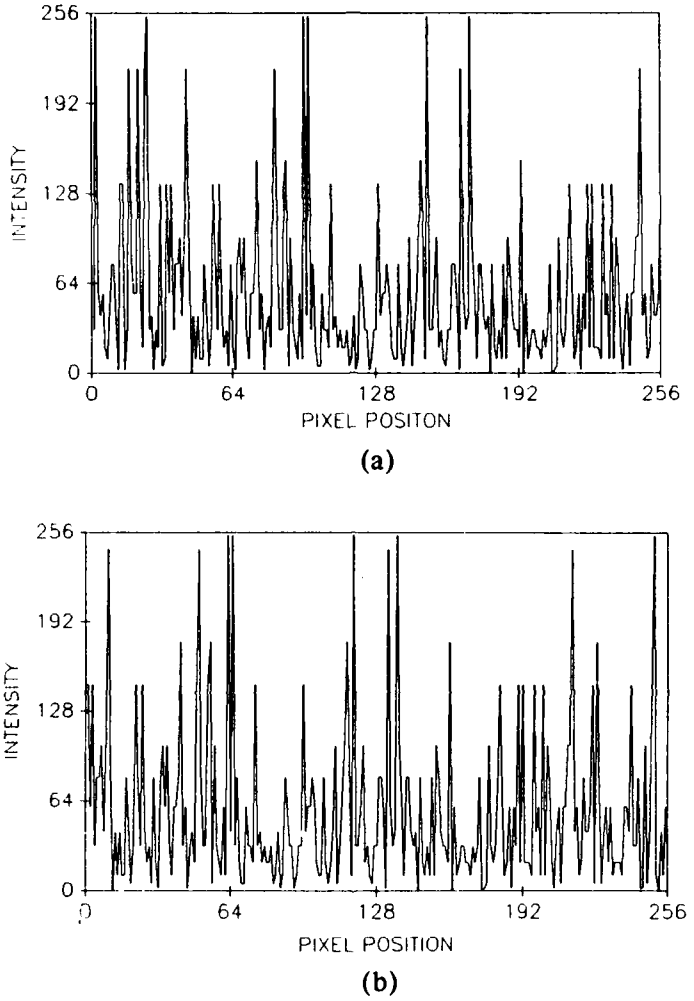


Fig. 7 — Profile of enhanced simulated speckle intensity
(a) before object rotation; (b) after object rotation

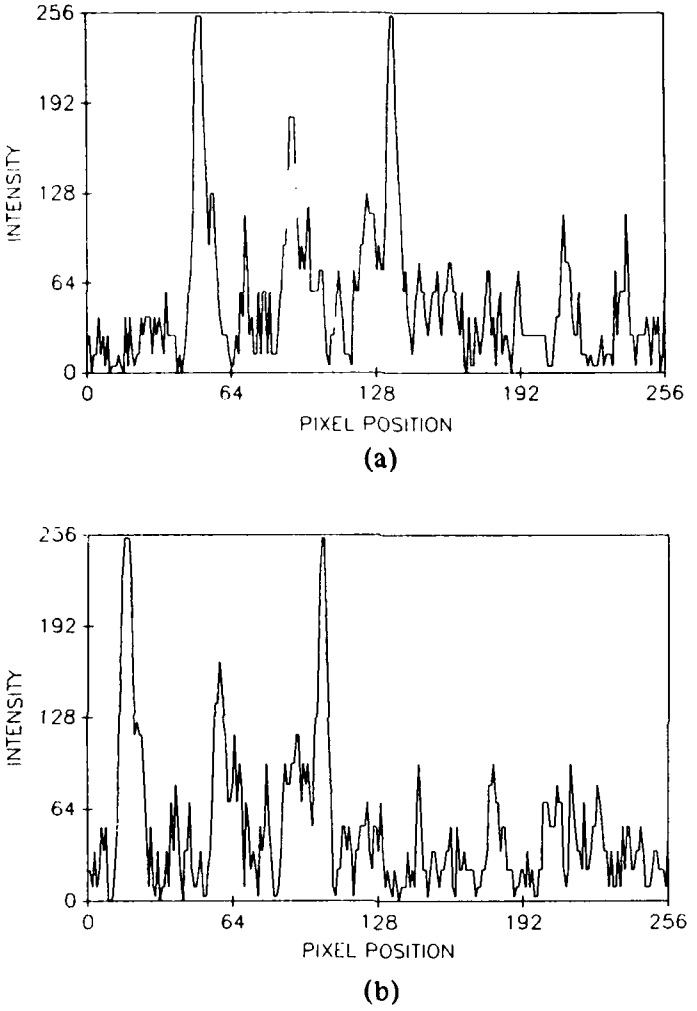


Fig. 8 — Profile of enhanced experimentally recorded speckle intensity
(a) before object rotation; (b) after object rotation

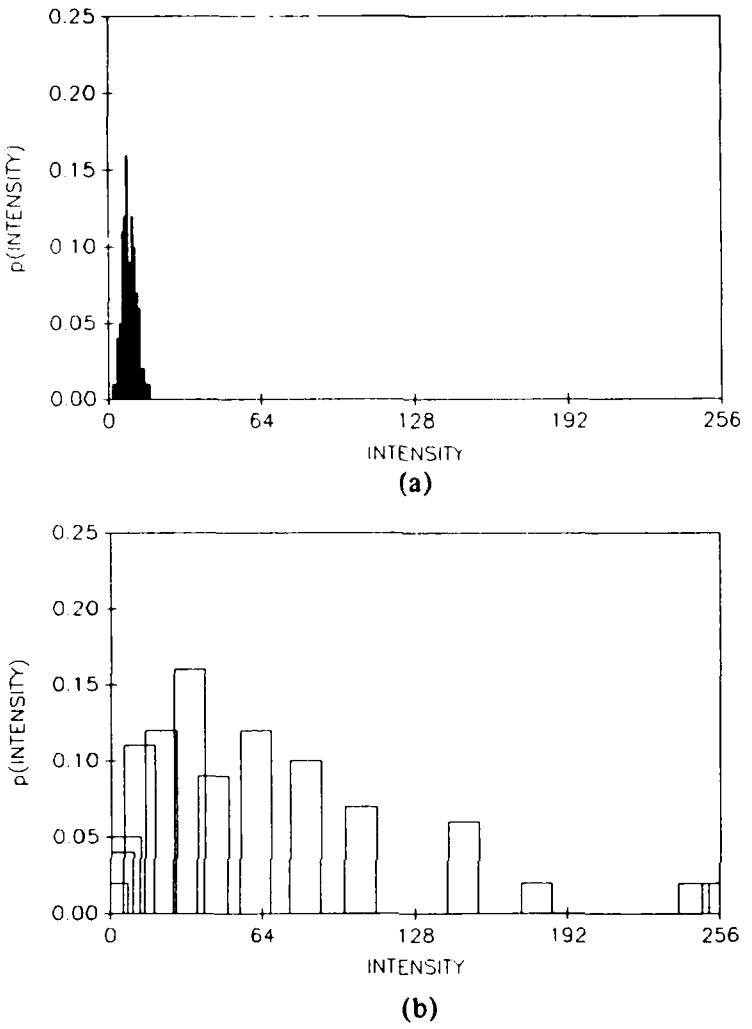


Fig. 9 — Probability density functions of simulated speckle intensity (a) before image enhamcement; (b) after enhancement

Figures 10(a) and 10(b) indicate the probability density functions for the experimentally recorded speckle. Note, however, that the probability density functions of the enhanced images (both simulated and experimental) are not perfect negative exponentials. This is because the variables in the transformations are not continuous, as we assumed in the derivation of the theory of the enhancement. The probability density function is only approximated by the histogram of the discrete levels.

To compute the magnitude of the cross correlation by direct evaluation, the 1-D arrays are slid over each other computing intensity products at each incremental shift. As the function is computed, the correlation information is multiplied by a triangular ramp of values equal to the number of overlapped elements at any given shifted position. To extract the correlation information, the resulting function must be divided by this triangle. The resulting values are normalized to the gray scale (0 to 255), and both the intensity and intensity profile of the correlation function are displayed on the display monitor.

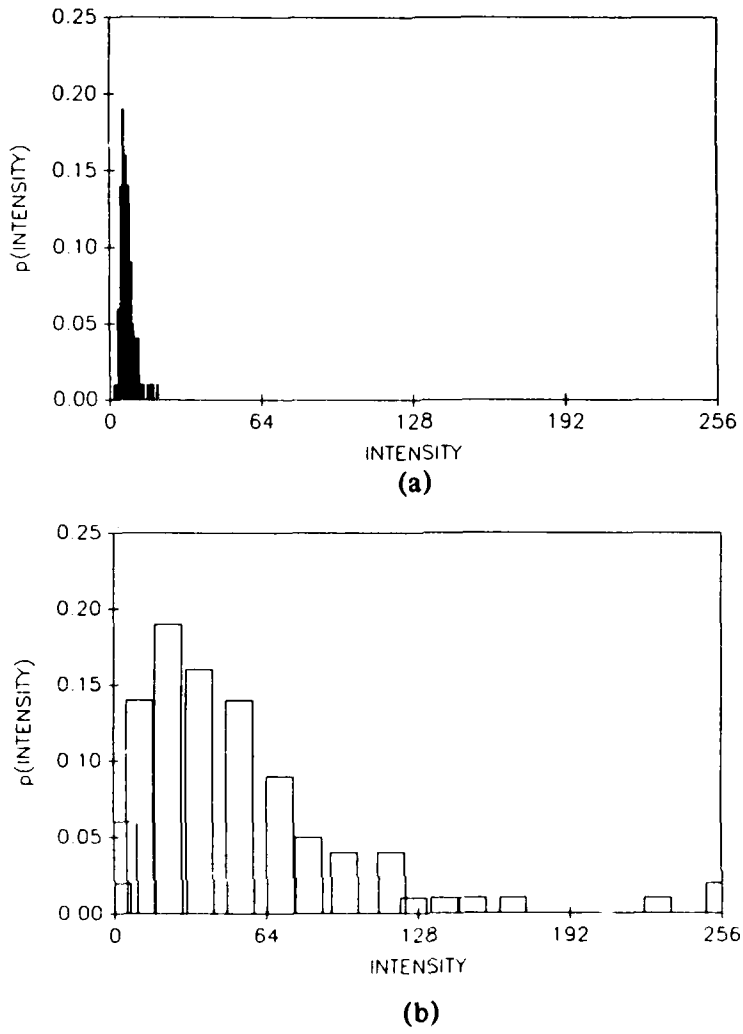


Fig. 10 — Probability density functions of experimentally recorded speckle intensity (a) before image enhancement; (b) after enhancement

Figures 11(a) and 11(b) show the magnitude of the cross-correlation function computed by direct evaluation for both the original simulated images and the enhanced simulated images. The horizontal axis is given as the position in millimeters on the detector surface from the optical axis. Since object rotation can be inferred from $x = 2\Delta\theta R$, and the correlation peak in Figs. 11(a) and 11(b) is at position x = 0.584 mm, the effective object rotation between samples for the simulated speckle is computed to be $\Delta\theta$ = 1.350 mrad.

An estimate of the SNR of the correlation function can be expressed in the same manner as the contrast, namely as the ratio of the standard deviation to the mean intensity. The SNR values for the functions shown in Figs. 11(a) and 11(b) are 0.02 and 0.10 respectively. The SNR ot the enhanced image is therefore improved by a factor of 5 for the direct evaluation of the cross-correlation function.

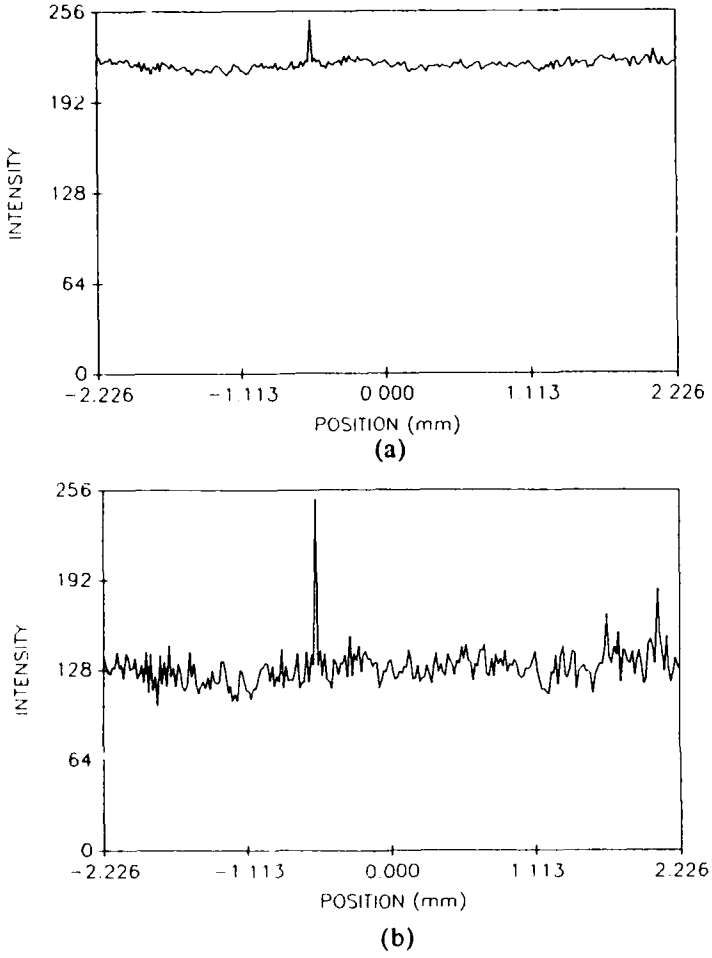


Fig. 11 — Magnitude of cross correlation computed by direct evaluation for images of (a) unenhanced and (b) enhanced simulated speckle

Figures 12(a) and 12(b) show this same function computed for the experimentally recorded images. The peak of this function occurs also at $x = 0.584$ mm, thus yielding the same value for $\Delta\theta$ as was found for the simulated speckle. During the experiment the recorded value of $\Delta\theta$ was 1.410 mrad. Comparing the recorded value of 1.410 mrad to the experimentally determined value of 1.350 mrad yields agreement within 5%.

The measurement of translation in one dimension gives an estimate for the rotation about the axis in the direction perpendicular to the sampling direction. Every effort was made to ensure perfect alignment, however, in the case of even a slight misalignment of the axis of rotation or of the detector, the rotation would be about an axis not parallel to the indicated axis. This condition produces speckle walk off, the translation of the speckle pattern in a direction not parallel with the original direction of sampling. This is a possible source of error in the measurements. The magnitude of this error could be determined by sampling in directions that make small angles with the principal direction to determine the maximum translation distance and the corresponding axis direction. Another approach would be to sample in the direction perpendicular to the principal direction to determine if there were any perpendicular translation. The translation distances in the two perpendicular directions would yield the magnitude and direction of the actual translation.

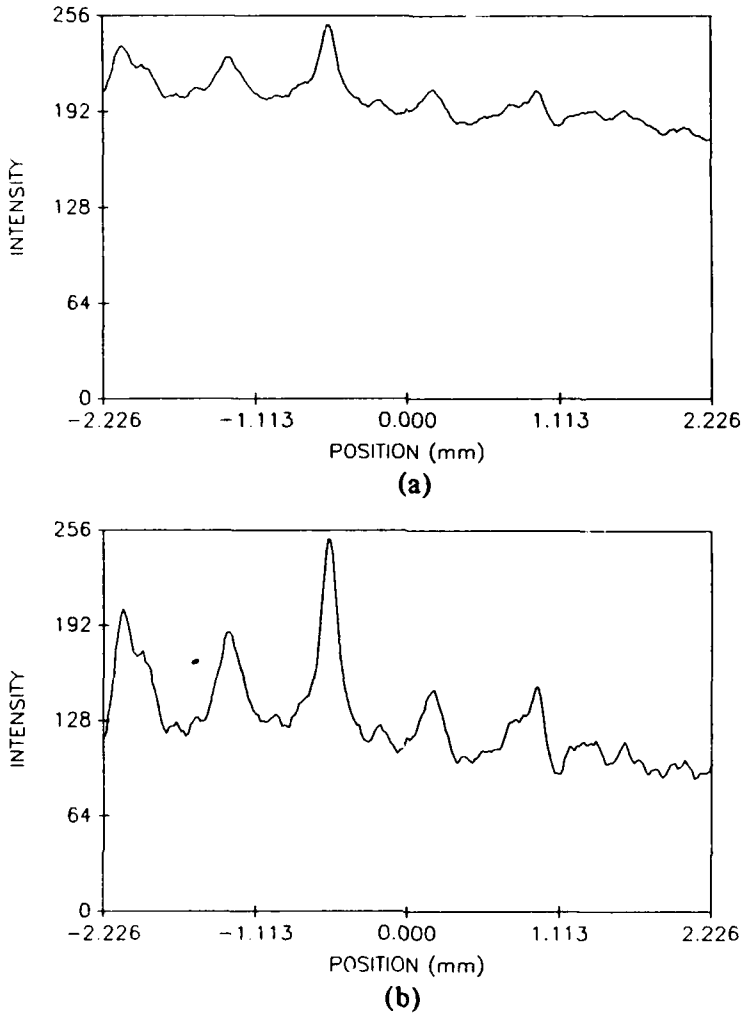


Fig. 12 — Magnitude of cross correlation computed by direct evaluation for images of (a) unenhanced and (b) enhanced experimentally recorded speckle

The SNRs for the direct evaluation of the cross correlation for the experimentally recorded images are 0.08 for the unenhanced images and 0.24 for the enhanced images, which is an increase of a factor of 3.

Fourier techniques are used to compute the magnitude of the normalized correlation function. The Fourier transform is performed by a subroutine that computes the FFT. The particular subroutine used is published by the IEEE Digital Signal Processing Committee of the IEEE Acoustics, Speech, and Signal Processing Society [17]. The image sample is imbedded in an array of zeros—enough zeros to double the length of the sample. This is necessary because if not buffered, as the arrays are slid over each other in the convolution process, they wrap around the ends of the samples and produce false correlation information.

To view the magnitudes of the FFT components so that the zero freqency component appears in the center of the array, the image function must be reordered. This is done automatically by multiplying the image function by the factor $(-1)^x$ prior to computing the FFT. When the magnitude of the FFT is

displayed as an image on the monitor, the magnitude values must be compressed since the dynamic range of the Fourier components is much greater than the dynamic range of the original image intensity values.

To compute Eq. (36), the numerator is determined by multiplying the FFT of the first sample with the complex conjugate of the FFT of the second sample and then taking the inverse FFT of the product. Marron and Schroeder [3] found that Eq. (36) was the least noisy when the denominator was computed by taking the expected value of each of the samples only over the overlap region of each of the samples. Thus, each of the values in the denominator of Eq. (36) is an array. These arrays are computed by cross correlating (again, by multiplying the Fourier transforms) each sample with a unit square pulse. Again, as with the direct evaluation, the triangle with which the information is multiplied must be divided by the numerator and denominator. Both numerator and denominator are arrays, so an element-by-element division is performed. The values for $|\mu_A|^2$ fall into the range $0 \leq |\mu_A|^2 \leq 1$, so they are scaled to the 0 to 255 gray scale, and both the intensity and the intensity profile are displayed.

Figures 13(a) and 13(b) show the magnitude of the normalized correlation coefficient of the original and enhanced sets of simulated images. Here the peak position of the correlation is also at 0.584 mm, the same value computed by direct correlation.

The SNRs for the correlation functions of the original and enhanced images is 1.38 and 1.59 respectively. Note that for the normalized correlation coefficient, enhancement produces a significantly smaller SNR improvement.

Figures 14(a) and 14(b) show the magnitude of the normalized correlation coefficient for the experimentally recorded images. The peak value of the normalized correlation coefficient is at 0.566 mm on the detector surface from the optical axis. As computed before from $x = 2\Delta\theta R$, it follows that $\Delta\theta = 1.310$ mrad. The accuracy of this result appears not to be as good as in the direct evaluation. This represents an error of 7%, slightly greater than the error computed earlier. The difference between the positions of the peaks is noted to be less than the width of either peak.

The SNRs for the two sets of normalized correlation coefficients in Figs. 14(a) and 14(b) are 1.63 for the correlation of original images and 1.88 for the correlation of the enhanced images. The SNRs for the normalized correlation coefficient, computed by Eq. (36), are well above those for the direct evaluation computations. The improvement in SNR for the normalized version of the cross correlation is on the order of a factor of 10. However the improvement in SNR that comes with enhancement is not nearly as substantial as the improvement in SNR that comes with the normalization of the correlation function.

Figure 15 shows the normalized correlation coefficient for the same object that has not rotated. This is the autocorrelation function, and the width of the function is an estimate of the average width of the speckle size. Note that from Figs. 13(a) and 13(b) the width of the cross-correlation function for the simulated speckle is one pixel, thus corresponding to the format the speckle was generated with.

Some limitations of the measurements should be addressed. The false correlation peaks at the periphery of the correlation functions are generated because the number of pixels entering the correlation sum decreases as the sampels are shifted past each other. When relatively few samples enter the correlation sum, false correlation peaks are generated at the edges of the function.

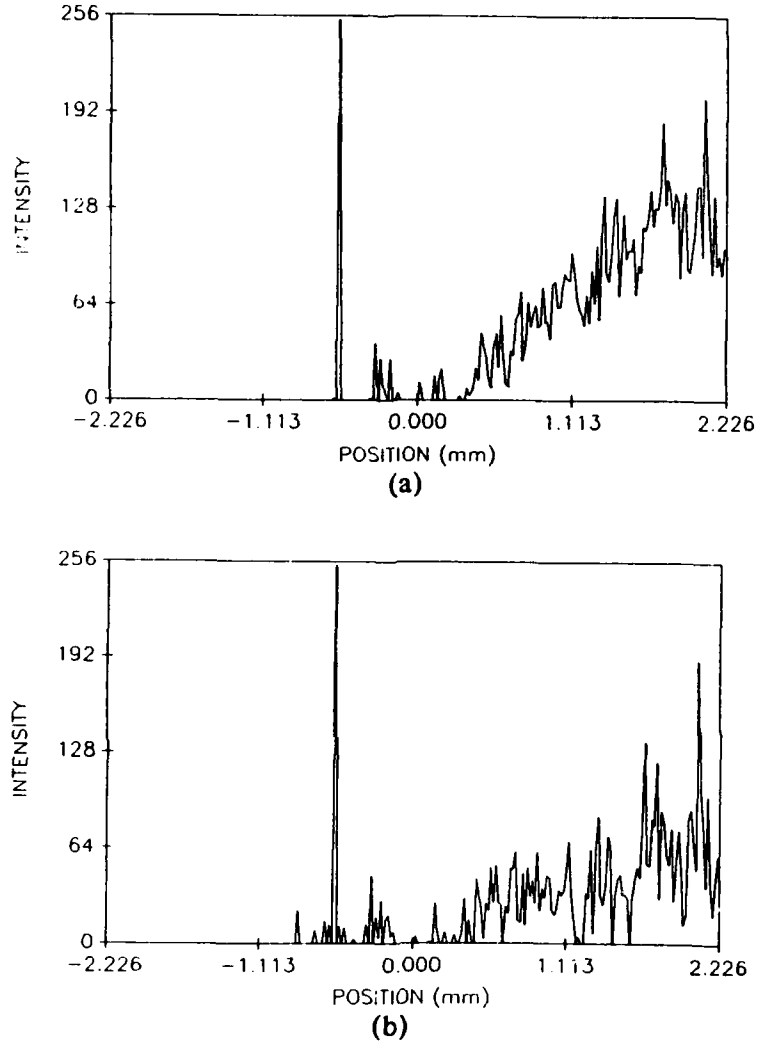
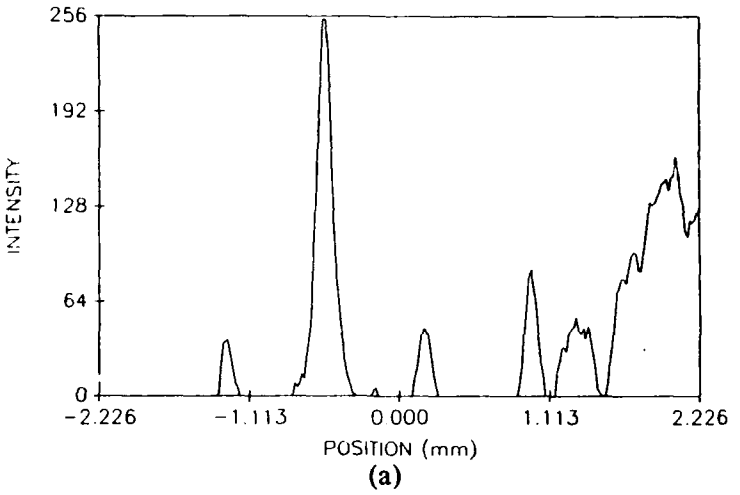


Fig. 13 — Magnitude of normalized correlation coefficient, $|\mu_A|^2$ computed by Fourier technique for (a) unenhanced and (b) enhanced simulated speckle



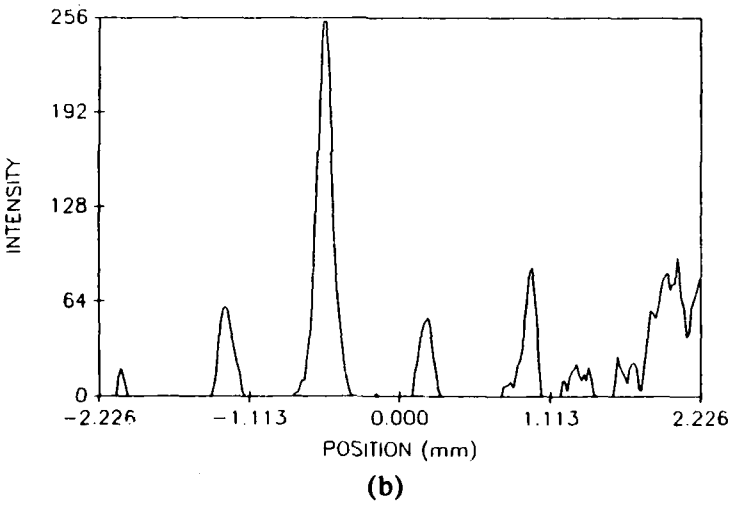


Fig. 14 — Magnitude of normalized correlation coefficient, $|\mu_A|^2$ computed by Fourier technique for (a) unenhanced and (b) enhanced experimentally recorded speckle

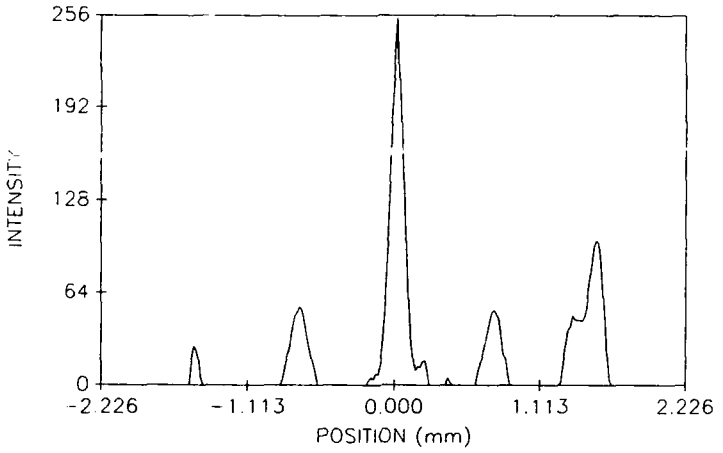


Fig. 15 — Autocorrelation function. Magnitude of $|\mu_A|^2$ for two experimentally recorded speckle images of a stationary object.

Some sources of error are attributed to the motion of the target, and some are attributed to the detection system. Speckle boiling is the degradation and generation of speckles that increases with the object rotation. Boiling can be attributed to the amount of object rotations and the placement of source and observation planes as well as underlying object shape [3]. Very little boiling was observed in the experimentally recorded data; however, evidence of boiling can be seen in Figs. 6(a) and 6(b) where the shapes of some of the peaks vary from the first sample in (a) to the second sample in (b).

Other sources of error include visible raster scan marks as can be seen in Fig. 4(b). Although these are visible to the human detector, they have negligible effect on the 1-D samples taken parallel to the marks. And although the recorded speckle did not show it, at different times during the experiment an interference pattern generated by the front and back surfaces of the quartz faceplate on the camera was superimposed on the speckle image pattern.

4. CONCLUSIONS

The results presented in this report demonstrate how 1-D speckle translation measurement is used to estimate object rotation. The technique involves recording speckle images before and after object rotation, computing the cross correlation of the two images, and inferring the rotation displacement from the computed translation distance. This report compares the unnormalized and normalized forms of the correlation function and demonstrates a technique in image enhancement that improves the image contrast for speckle with less than ideal statistical properties.

We can make estimates of the amount of computing time that 1-D processing saves. For example, the computation time of an FFT is proportional to ($N \log_2 N$), so by increasing the number of samples from N to N^2 also requires increasing the processing time by the factor $2N$. Computation time is conserved whenever a 2-D $N \times N$ array is traded for a line sample of N elements as input to an equation such as Eq. (36).

Computation of correlation functions for 1-D samples is subject to the error that results when a finite number of intensity samples are used to compute Eq. (36) [11]. Thus, the fewer the intensity samples correlated, the smaller the SNR of the correlation function. The quality of the functions from

1-D samples is inherently limited by the size of the image. Two dimensional sampling increases the number of intensity samples in the computation of Eq. (36), however, it puts demands on computer memory and processing time.

By sampling in only one direction, the only object rotation that can be estimated is rotation about the single axis perpendicular to the direction of sampling. To detect rotation about any other axis, 1-D samples at other orientations are necessary. A useful extension to this project would be to sample in two perpendicular directions. Thus, rotations about the two perpendicular axes could be resolved. As an example, consider a rotation about an axis that makes a 45° angle with the axis perpendicular to the direction of sampling. The component of the rotation about this axis in the direction of the axis sampled is measured, but with restrictions. Once a given speckle has moved off the line of sampling the subsequent image will not record that speckle, and the two images decorrelate. Therefore, this approach would be limited to the measurement of translation distances the size of a speckle.

Comparisons may be made between the unnormalized and normalized versions of the correlation function. For the example calculations, the normalized form had a SNR of a factor of 10 higher than the SNR for the unnormalized function. Clearly, the normalization of the correlation function sharpens the peak of the function and reduces the magnitude of the noise associated with the function.

For the computation of the cross correlation for original and enhanced imagery, the enhanced imagery always has a higher SNR. For the normalized correlation coefficient, the ratio of SNRs was approximately 1.2. The improvement here is not as drastic as the SNR improvement associated with normalization, however, a visual inspection of the plots (see Figs. 14(a) and 14(b)) shows a noticeable improvement.

To consider possibilities for further work in this direction, the meaurement of real-time rotation should be mentioned. The method of still-life image capture used in this investigation only estimates the rotation displacement over some given duration. Investigations of the feasibility of measuring real-time rotation rates are currently ongoing. One investigation involves the use of a pulsed laser to illuminate a rotating target. The laser pulse serves to strobe the speckle pattern onto the detector. Perhaps techniques involving the use of a high speed camera or camera shutter system could also provide the imagery necessary to determine the rotation rate of an object. The fastest rotation rate measurable is limited only by the time it takes to capture an image.

7. REFERENCES

1. N. George, "Speckle from Rough, Moving Objects," *J. Opt. Soc. Am.* **66**, 1182 (1976).

2. A. Hayashi and Y. Kitagawa, "High-Resolution Rotation-Angle Measurement of a Cylinder Using Speckle Displacement Detection," *Appl. Opt.* **22**, 3520 (1983).

3. J. Marron and K. Schroeder, "Speckle from Rough Rotating Objects," *Appl. Op.* **27**, 4279 (1988).

4. H. Fujii and T. Asakura, "Effect of Surface Roughness on the Statistical Distribution of Image Speckle Intensity," *Opt. Comm.* **11**, 35 (1974).

5. M. Francon, *Laser Speckle and Applications in Optics* (Academic Press, New York, 1979).

6. J. W. Goodman, "Statistical Properties of Laser Speckle Patterns," in *Laser Speckle and Related Phenomena,* J. C. Dainty, ed. (Springer-Verlag, Berlin, 1975).

7. W. B. Davenport, Jr., and W. L. Root, *An Introduction to the Theory of Random Signals and Noise* (McGraw-Hill, New York, 1958).

8. J. W. Goodman, *Introduction to Fourier Optics* (McGraw-Hill, New York, 1968).

9. J. W. Goodman, Proc. IEEE **53**, 1688 (1965).

10. M. Born and E. Wolf, *Principles of Optics* (Pergamon, Oxford, 1975).

11. J. Marron, "Accuracy of Fourier Magnitude Estimation from Speckle Intensity Correlation," *J. Opt. Soc. Am.* **5**, 864 (1988).

12. R. A. Sprague, "Surface Roughness Measurement Using White Light Speckle," *Appl. Opt.* **11**, 2811 (1972).

13. J. C. Leader, "An Analysis of the Frequency Spectrum of Laser Light Scattered from Moving Rough Objects," *J. Opt. Soc. Am.* **67**, 1091 (1977).

14. R. C. Gonzalez and R. Wintz, *Digital Imaging Processing* (Addison-Wesley, London, 1977), Chap. 4.

15. *ImageTool™ Toolkit for Image Processing Software Development*, 1985 Santa Monica, CA: Werner Frei Associates.

16. W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, *Numerical Receipes* (Cambridge University Press, Cambridge, 1986), Chap. 7.

17. Digital Signal Processing Committee IEEE Acoustics, Speech, and Signal Processing Soceity, Ed., *Programs for Digital Signal Processing* (IEEE Press, New York, 1979), Chap. 1.

Appendix A

```
      PROGRAM SPECSIM
C     BL 12 FEB 1990
C     MASTER'S THESIS 1990
C     PROGRAM GENERATES SPECKLE WITH EXPONENTIAL STATISTICS
C     ALSO SPECKLE WITH POISSONIAN DISTRIBUTION
C     FUNCTIONS POIDEV, EXPDEV, RAN1, GAMMLN PUBLISHED IN
C     NUMERICAL RECIPES, BY FLANNERY, et al.
      INTEGER*2   RNPTS, CNPTS
      INTEGER*2   TRANS, Y, GAIN
      CHARACTER*16      FNAME
      PARAMETER (RNPTS= 256, CNPTS= 256)
      DIMENSION Y(65536)
      CALL SETUP
      CALL DIGITZ (1)
1     WRITE(*,*)'CHOOSE (1)EXPONENTIAL, OR (2)POISSON DISTRIBUTION'
      READ (*, *) DISTR
      IF (DISTR.EQ.2) THEN
            WRITE (*, *) ' ENTER VALUE FOR MEAN: '
            READ (*, *) XM
      ENDIF
      WRITE (*, *) ' ENTER VALUE FOR GAIN: '
      READ (*, *) GAIN
      IDUM= -1
      DO 11 I= 1, (RNPTS*CNPTS)
      IF (DISTR.EQ.1) Y(I)= EXPDEV(IDUM) * GAIN
      IF (DISTR.EQ.2) Y(I)= POIDEV(XM, IDUM) * GAIN
11    CONTINUE
      CALL WRREC (128, 128, CNPTS, RNPTS, 0, Y)
      WRITE (*, *) ' FILE NAME: '
      READ (*, 101) FNAME
      CALL CHARZ (FNAME, 16)
      CALL WDISK (IER, FNAME, 128, 128, CNPTS, RNPTS)
C     TRANSLATE SPECKLE PATTERN
      WRITE (*, *) ' PIXELS TRANSLATED: '
      READ (*, *) TRANS
      DO 12 I=1, ((RNPTS*CNPTS)-TRANS)
      Y(I)= Y(I+TRANS)
12    CONTINUE
      CALL WRREC (128, 128, CNPTS, RNPTS, 0, Y)
      WRITE (*, *) ' FILE NAME: '
      READ (*, 101) FNAME
      CALL CHARZ (FNAME, 16)
      CALL WDISK (IER, FNAME, 128, 128, CNPTS, RNPTS)
101   FORMAT (A16)
      GOTO 1
      END
C     ************************************************
      FUNCTION EXPDEV(IDUM)
      EXPDEV= -LOG (RAN1(IDUM))
      RETURN
```

```
      END
C     ************************************************
      FUNCTION RAN1(IDUM)
      DIMENSION R(97)
      PARAMETER (M1=259200,IA1=7141,IC1=54773,RM1=3.8580247E-6)
      PARAMETER (M2=134456,IA2=8121,IC2=28411,RM2=7.4373773E-6)
      PARAMETER (M3=243000,IA3=4561,IC3=51349)
      DATA IFF /0/
      IF (IDUM.LT.0.OR.IFF.EQ.0) THEN
       IFF=1
       IX1=MOD(IC1-IDUM,M1)
       IX1=MOD(IA1*IX1+IC1,M1)
       IX2=MOD(IX1,M2)
       IX1=MOD(IA1*IX1+IC1,M1)
       IX3=MOD(IX1,M3)
       DO 11 J=1,97
        IX1=MOD(IA1*IX1+IC1,M1)
        IX2=MOD(IA2*IX2+IC2,M2)
        R(J)=(FLOAT(IX1)+FLOAT(IX2)*RM2)*RM1
11     CONTINUE
       IDUM=1
      ENDIF
      IX1=MOD(IA1*IX1+IC1,M1)
      IX2=MOD(IA2*IX2+IC2,M2)
      IX3=MOD(IA3*IX3+IC3,M3)
      J=1+(97*IX3)/M3
      IF(J.GT.97.OR.J.LT.1)PAUSE
      RAN1=R(J)
      R(J)=(FLOAT(IX1)+FLOAT(IX2)*RM2)*RM1
      RETURN
      END
C     ************************************************
      SUBROUTINE SETUP
      INTEGER*2   NBRDS, BCK( 1), ZERO( 1)
      INTEGER*2   RAMP (256), ZEROES(256)
      INTEGER*4   MEMSEG(1), CTLREG(1)
      DATA        ZEROES/256*0/, BCK/65/, ZERO/0/
C     REQUIRED INITIALIZATION
      NBRDS= 1
      MEMSEG (1)= 851968 / 16
      CTLREG(1)= 65280
      CALL INITIM (1, MEMSEG, CTLREG)
      CALL CLKSEL (1)
      CALL BRDSEL (1)
      DO 100 I=1, 256
      RAMP(I)= I-1
100   CONTINUE
      CALL LUTSEL (12)
      CALL WRLUT (0, 256, RAMPL)
      CALL LUTSEL (0)
      CALL WRLUT (0, 256, RAMP)
      CALL WRLUT (255, 1, ZERO)
```

```
      CALL WRLUT (0, 1, BCK)
      CALL LUTSEL (4)
      CALL WRLUT (0, 256, RAMP)
      CALL LUTSEL (8)
      CALL WRLUT (0, 256, RAMP)
      CALL WRLUT (255, 1, ZERO)
      RETURN
      END
C     ************************************************************
      FUNCTION POIDEV(XM,IDUM)
      PARAMETER (PI=3.141592654)
      DATA OLDM /-1./
      IF (XM.LT.12.)THEN
       IF (XM.NE.OLDM) THEN
        OLDM=XM
        G=EXP(-XM)
       ENDIF
       EM=-1
       T=1.
2      EM=EM+1.
       T=T*RAN1(IDUM)
       IF (T.GT.G) GO TO 2
      ELSE
       IF (XM.NE.OLDM) THEN
        OLDM=XM
        SQ=SQRT(2.*XM)
        ALXM=ALOG(XM)
        G=XM*ALXM-GAMMLN(XM+1.)
       ENDIF
1      Y=TAN(PI*RAN1(IDUM))
       EM=SQ*Y+XM
       IF (EM.LT.0.) GO TO 1
       EM=INT(EM)
       T=0.9*(1.+Y**2)*EXP(EM*ALXM-GAMMLN(EM+1.)-G)
       IF (RAN1(IDUM).GT.T) GO TO 1
      ENDIF
      POIDEV=EM
      RETURN
      END
C     ************************************************************
      FUNCTION GAMMLN(XX)
      REAL*8 COF(6),STP,HALF,ONE,FPF,X,TMP,SER
      DATA COF,STP/76.18009173D0,-86.50532033D0,24.01409822D0,
     *   -1.231739516D0,.120858003D-2,-.536382D-5,2.50662827465D0/
      DATA HALF,ONE,FPF/0.5D0,1.0D0,5.5D0/
      X=XX-ONE
      TMP=X+FPF
      TMP=(X+HALF)*LOG(TMP)-TMP
      SER=ONE
      DO 11 J=1,6
       X=X+ONE
       SER=SER+COF(J)/X
```

```
11   CONTINUE
     GAMMLN=TMP+LOG(STP*SER)
     RETURN
     END


C      ***************************************************************
C      ***************************************************************


C    PROGRAM SPECSTO
C    BL 12/89
C    IMAGE PROCESSING CODE
C    STORES IMAGERY TO DISK FOR LATER RETRIEVAL
C
C
C    COMPILED WITH MICROSOFT FORTRAN77 3.30
C    AND LIBRARY: IMTOOL BY W. FREI
C
C
     INTEGER*2  RESPONSE
     INTEGER*2  ICOL, IROW
     INTEGER*4  NCOL, NROW


     CALL SETUP
10   CALL MENU (RESPONSE)
     IF (RESPONSE.EQ.1) CALL ACQUIRE (ICOL, IROW, NCOL, NROW)
     IF (RESPONSE.EQ.2) CALL HISTO
     IF (RESPONSE.EQ.9) STOP
     GOTO 10
99   STOP
     END
C      ********************************************************
       SUBROUTINE SETUP
       INTEGER*2  NBRDS, BCK( 1), ZERO( 1)
       INTEGER*2  RAMP (256), ZEROES(256)
       INTEGER*4  MEMSEG(1), CTLREG(1)
       DATA       ZEROES/256*0/, BCK/65/, ZERO/0/
C      REQUIRED INITIALIZATION
       NBRDS= 1
       MEMSEG (1)= 851968 / 16
       CTLREG(1)= 65280
       CALL INITIM (1, MEMSEG, CTLREG)
       CALL CLKSEL (1)
       CALL BRDSEL (1)
       DO 100 I=1, 256
       RAMP(I)= I-1
100    CONTINUE
       CALL LUTSEL (12)
       CALL WRLUT (0, 256, RAMPL)
       CALL LUTSEL (0)
```

```
      CALL WRLUT (0, 256, RAMP)
      CALL WRLUT (255, 1, ZERO)
      CALL WRLUT (0, 1, BCK)
      CALL LUTSEL (4)
      CALL WRLUT (0, 256, RAMP)
      CALL LUTSEL (8)
      CALL WRLUT (0, 256, RAMP)
      CALL WRLUT (255, 1, ZERO)
      RETURN
      END
C     ****************************************
      SUBROUTINE MENU ( RESPONSE)
      INTEGER*2  RESPONSE
99    WRITE (*, 1000)
      READ (*, 1001) RESPONSE
      IF (RESPONSE.EQ.1) GOTO 100
      IF (RESPONSE.EQ.2) GOTO 100
      IF (RESPONSE.EQ.9) GOTO 100
      GOTO 99
1000  FORMAT (TR15, ' CHOOSE ONE OF THE FOLLOWING: ',
     & //TR20, ' 1. STORE SERIES OF FRAMES',
     & //TR20, ' 2. PERFORM HISTOGRAM ENHANCEMENT',
     & //TR20, ' 9. EXIT TO DOS')
1001  FORMAT (I3)
100   RETURN
      END
C     ****************************************
      SUBROUTINE ACQUIRE (ICOL, IROW, NCOL, NROW)
C
C     ROUTINE SAVES IMAGES TO DISK FOR LATER RETREVEAL
C
      INTEGER*2  WHICH, IER, I
      INTEGER*2  ICOL, IROW
      INTEGER*4  NCOL, NROW
      CHARACTER*13      FNAME
      CALL DIGITZ (2)
100   WRITE (*, 1030)
      READ (*, 1031) WHICH
      IF (WHICH.EQ.1) GOTO 99
      GOTO 100
99    CONTINUE
      PAUSE 'FILES WILL BE WRITTEN TO DIRECTORY \IMG'
      DO 85 I= 1, 9
      PAUSE 'HIT RETURN WHEN DESIRED FRAME IS ON MONITOR'
      CALL DIGITZ (1)
      IF (WHICH.EQ.1) THEN
      ICOL= 128
      IROW= 128
      NCOL= 256
      NROW= 256
      END IF
      IF (I.EQ.1) FNAME= 'IMG\IMG11'
```

```
      IF (I.EQ.2) FNAME= 'IMG\IMG12'
      IF (I.EQ.3) FNAME= 'IMG\IMG13'
      IF (I.EQ.4) FNAME= 'IMG\IMG14'
      IF (I.EQ.5) FNAME= 'IMG\IMG15'
      IF (I.EQ.6) FNAME= 'IMG\IMG16'
      IF (I.EQ.7) FNAME= 'IMG\IMG17'
      IF (I.EQ.8) FNAME= 'IMG\IMG18'
      IF (I.EQ.9) FNAME= 'IMG\IMG19'
      CALL CHARZ (FNAME, 13)
      CALL WDISK (IER, FNAME, ICOL, IROW, NCOL, NROW)
85    CONTINUE
1030  FORMAT (TR15, ' ACQUISITION MENU- CHOOSE ONE OF THE
FOLLOWING: ',
     & //TR20, ' 1. WHOLE FRAME')
1031  FORMAT (I2)
2002  FORMAT (A4)
2005  FORMAT (I1)
2006  FORMAT (A1)
      RETURN
      END
C     ************************************************************
      SUBROUTINE HISTO
      REAL        ARRAY, NARRAY, REAPIX
      REAL        AVG, AVERAG, STANDV, CONTRA
      INTEGER*2   PIXELS, PIXELA
      CHARACTER*16          DFNAME, FNAME
      DIMENSION   ARRAY( 256), PIXELA( 65536), REAPIX( 65536)
      DIMENSION       NARRAY( 256), PIXELS( 256)
C     OPEN FILE CONTAINING IMAGE
      WRITE (*, *) 'IMAGE FILE NAME: '
      READ (*, 1032) DFNAME
      CALL CHARZ ( DFNAME, 16)
      CALL DIGITZ ( 0)
      CALL RDISK (IER, DFNAME, 128, 128, 256, 256)
      CALL RDREC (128, 256, 256, 1, 0, PIXELS)
      DO 6 I= 1, 256
      ARRAY( I)= 0.0
6     CONTINUE
      DO 8 I=1, 256
      ARRAY( PIXELS( I)+1)= ARRAY( PIXELS( I)+1) + 1
8     CONTINUE
      DO 9 I=2, 256
      ARRAY( I)= ARRAY( I-1) + ARRAY( I)
9     CONTINUE

      DO 10 I= 1, 256
      REAPIX( I)= (ARRAY( PIXELS( I) + 1)) / 256
10    CONTINUE
      AVG= 20
12    NARRAY( 1)= (EXP( -1/AVG)) / AVG
      DO 20 I=2, 256
      NARRAY( I)= NARRAY(I-1) + (EXP(-I / AVG) / AVG)
```

```
20   CONTINUE
     DO 50 I=1, 256
     DO 40 J=1, 256
     IF (REAPIX( I).LE.NARRAY( J)) THEN
      PIXELS( I)= ABS( J-1)
      GOTO 50
     ENDIF
     IF (REAPIX( I).GE.NARRAY( 256)) THEN
      PIXELS( I)= 254
      GOTO 50
     ENDIF
40   CONTINUE
50   CONTINUE
     AVERAG= 0.0
     STANDV= 0.0
     CONTRA= 0.0
     DO 55 I=1, 256
C    COMPUTE CONTRAST AND INCREASE AVERAGE IF NECESSARY
     AVERAG= AVERAG + PIXELS( I)
55   CONTINUE
     AVERAG= AVERAG / 256
     DO 57 I=1, 256
     STANDV= (PIXELS( I) - AVERAG)**2 + STANDV
57   CONTINUE
     STANDV= SQRT (STANDV / 256)
     CONTRA= STANDV / AVERAG
     IF ((CONTRA.GT.(1.0)).OR.(CONTRA.LT.(0.95))) THEN
     AVG= AVG + 1
     GOTO 12
     ENDIF
     CALL RDREC( 128, 128, 256, 256, 0, PIXELA)
     DO 108 I=1, 65536
     ARRAY( PIXELA( I)+1)= ARRAY( PIXELA( I)+1) + 1
108  CONTINUE
     DO 109 I= 2. 256
     ARRAY( I)= ARRAY( I-1) + ARRAY( I)
109  CONTINUE
     DO 110 I=1, 65536
     REAPIX( I)= (ARRAY( PIXELA( I) + 1)) / 65536
110  CONTINUE
     DO 150 I=1, 65536
     DO 140 J=1, 256
     IF (REAPIX( I).LE.NARRAY( J)) THEN
               PIXELA( I)= ABS( J-1)
               GOTO 150
               ENDIF
     IF (REAPIX( I).GE.NARRAY( 256)) THEN
               PIXELA( I)= 254
               GOTO 150
               ENDIF
140  CONTINUE
150  CONTINUE
```

```
      CALL WRREC (128, 128, 256, 256, 0, PIXELA)
      CALL WRREC (128, 256, 256, 1, 0, PIXELS)
      WRITE (*, *) 'STORE IMAGE IN FILE: '
      READ (*, 1032) FNAME
      CALL CHARZ ( FNAME, 16)
      CALL WDISK (IER, FNAME, 128, 128, 256, 256)
 1032 FORMAT( A16)
      RETURN
      END


C     **********************************************************************
C     **********************************************************************




C     PROGRAM SPECPRO
C     BL 7/89
C     IMAGE PROCESSING CODE
C     SPECKLE FROM ROTATING OBJECT
C
C     CODE FOR 1D PROCESSING 9/89
C
C     COMPILED WITH MICROSOFT FORTRAN77 3.30
C     LINK CODE WITH OBJ FILES: FFT
C     AND LIBRARY: IMTOOL BY W. FREI
C
C     MENU DRIVEN FORMAT FOR DIFFERENT PROGRAM SEGMENTS
C     2DFFT ADAPTED FROM IEEE PUBLICATION, SEE SUBROUTINE
C     HEADER
C
      INTEGER*2  RESPONSE
      INTEGER*2  ICOL, IROW
      INTEGER*4  NCOL, NROW
      REAL       A, B, C, D, UNIT, ZEROS
      REAL       TRIPEAK
      INTEGER*2  GETPIX
      COMMON          A, B, C, D, UNIT, ZEROS
      COMMON          GETPIX

 10   CALL MENU (RESPONSE)
      IF (RESPONSE.EQ.1.OR.RESPONSE.EQ.2) THEN
      CALL SETUP
      CALL ACQUIRE (ICOL, IROW, NCOL, NROW, RESPONSE)
     END IF
      IF (RESPONSE.EQ.3) THEN
      CALL XFORM (ICOL, IROW, NCOL, NROW, TRIPEAK)
      CALL XCORR (NCOL, NROW, TRIPEAK)
      ENDIF
```

```
C       IF (RESPONSE.EQ.4) CALL XCORR (NCOL, NROW, TRIPEAK)

        IF (RESPONSE.EQ.9) GOTO 99
        GOTO 10
99      STOP
        END
C       ****************************************
        SUBROUTINE SETUP
        INTEGER*2   NBRDS, BCK( 1), ZERO( 1)
        INTEGER*2   RAMP (256), ZEROES (256)
        INTEGER*4   MEMSEG (1), CTLREG (1)
        DATA        ZEROES/ 256*0/, BCK/ 65/, ZERO/ 0/
C       REQUIRED INITIALIZATION
        NBRDS= 1
        MEMSEG (1)= 851968 / 16
        CTLREG (1)= 65280
C
        CALL INITIM (1, MEMSEG, CTLREG)
        CALL CLKSEL (1)
        CALL BRDSEL (1)
C
        DO 100 I=1, 256
        RAMP (I)= I - 1
100     CONTINUE
        CALL LUTSEL (12)
        CALL WRLUT (0, 256, RAMP)
        CALL LUTSEL (0)
        CALL WRLUT (0, 256, RAMP)
        CALL WRLUT (255, 1, ZERO)
        CALL WRLUT (0, 1, BCK)
        CALL LUTSEL (4)
        CALL WRLUT (0, 256, RAMP)

        CALL LUTSEL (8)
        CALL WRLUT (0, 256, RAMP)
        CALL WRLUT (255, 1, ZERO)
C
        CALL DIGITZ (2)
        RETURN
        END
C       ****************************************
        SUBROUTINE MENU ( RESPONSE)
        INTEGER*2   RESPONSE
99      WRITE (*, 1000)
        READ (*, 1001) RESPONSE
        IF (RESPONSE.EQ.1) GOTO 100
        IF (RESPONSE.EQ.2) GOTO 100
        IF (RESPONSE.EQ.3) GOTO 100
C       IF (RESPONSE.EQ.4) GOTO 100
        IF (RESPONSE.EQ.9) GOTO 100
        GOTO 99
1000    FORMAT (TR15, ' MAIN MENU- CHOOSE ONE OF THE FOLLOWING: ',
```

```
     &  //TR20, ' 1. ACQUIRE TWO IMAGES'
     &  //TR20, ' 2. GET TWO IMAGES FROM DISK'
     &  //TR20, ' 3. VIEW CORRELATION FUNCTIONS'
     &  //TR20, ' 9. EXIT TO DOS')
1001   FORMAT (I3)
100    RETURN
       END
C      ****************************************
       SUBROUTINE ACQUIRE (ICOL, IROW, NCOL, NROW, RESPONSE)
C
C      ROUTINE GETS ONE LINE FROM VIDEO AND WRITES PIXEL VALUES
C      TO
C      DISK FOR PROCESSING
C
       INTEGER*2   RESPONSE
       INTEGER*2   ICOL, IROW, DISCOL, DISROW
       INTEGER*4   NCOL, NROW, DINCOL, DINROW
       CHARACTER*6       FNAME
       CHARACTER*16      DFNAME

       IF (RESPONSE.EQ.1) THEN
       CALL DIGITZ (2)
       PAUSE 'HIT RETURN WHEN DESIRED FRAME IS ON MONITOR'
       CALL DIGITZ (1)
       CALL AREAPR
       END IF
       ICOL= 128
       IROW= 255
       NCOL= 256
       NROW= 1
       IF (RESPONSE.EQ.2) THEN
       CALL SETSCR (0)
       DO 98 I=1, 2
       IF (I.EQ.1) FNAME= 'SPEC1'
       IF (I.EQ.2) FNAME= 'SPEC2'
       WRITE (*, *) 'IMAGE FILE NAME: '
       READ (*, 1032) DFNAME
       CALL CHARZ (DFNAME, 16)
       CALL CHARZ (FNAME, 6)
       DISCOL=128
       DISROW=128
       DINCOL=256
       DINROW=256
       CALL DIGITZ (0)
       CALL RDISK (IER, DFNAME, DISCOL, DISROW, DINCOL, DINROW)
       CALL AREAPR
       CALL OUTLINE (ICOL, IROW, NCOL, NROW)
       CALL WDISK (IER, FNAME, ICOL, IROW, NCOL, NROW)
98     CONTINUE
       END IF
       IF (RESPONSE.EQ.1) THEN
       FNAME= 'SPEC1'
```

```
      CALL CHARZ (FNAME, 6)
      CALL OUTLINE (ICOL, IROW, NCOL, NROW)
      CALL WDISK (IER, FNAME, ICOL, IROW, NCOL, NROW)
      PAUSE 'HIT RETURN WHEN 2ND FRAME IS ON MONITOR'
      CALL DIGITZ (1)
      CALL AREAPR
      FNAME= 'SPEC2'
      CALL CHARZ (FNAME, 6)
      CALL OUTLINE (ICOL, IROW, NCOL, NROW)
      CALL WDISK (IER, FNAME, ICOL, IROW, NCOL, NROW)
      END IF
1032  FORMAT (A16)
      RETURN
      END
C     ****************************************
      SUBROUTINE OUTLINE (ICOL, IROW, NCOL, NROW)
C     ROUTINE OUTLINES CHOSEN AREA
      INTEGER*2  VALUE
      INTEGER*2  XSTART, YSTART, XEND, YEND
      INTEGER*2  ICOL, IROW
      INTEGER*4  NCOL, NROW
      DATA VALUE /250/
      XSTART= ICOL
      YSTART= IROW - 1
      XEND= NCOL + XSTART
      YEND= YSTART
      CALL DWVEC (XSTART, YSTART, XEND, YEND, VALUE)
      RETURN
      END
C     ****************************************
      SUBROUTINE XFORM (ICOL, IROW, NCOL, NROW, TRIPEAK)
C     ROUTINE PLACES STORED IMAGES TO SCREEN
C     TAKES FOURIER XFORM OF IMAGES, PLACES ALSO ON SCREEN
      CHARACTER*6      FNAME
      INTEGER*2  ICOL, IROW
      INTEGER*4  NCOL, NROW, SIZE
      REAL       A, B, C, D, UNIT, ZEROS, RHOLD, PIX
      REAL       STD, AVG, TRIPEAK, CONTRS
      REAL*4           GAIN
      INTEGER*2  GETPIX
      INTEGER*2  INTNUM
      INTEGER*2  IER, IC, IR, FLAG, ICFFT, IRFFT
      INTEGER          TEMP, IMIN, IMAX
      DOUBLE PRECISION DSTACK(2500)
      COMMON /CSTAK/ DSTACK
      COMMON           A, B, C, D, UNIT, ZEROS
      COMMON           GETPIX
      DIMENSION  A(512), B(512), GETPIX(512), PIX(512)
      DIMENSION  C(512), D(512)
      DIMENSION  UNIT(512), ZEROS(512)
      DATA       INTNUM/0/, SIZE/512/
      CALL SETSCR (INTNUM)
```

```
      FLAG= 1
      DO 10 I=1, SIZE
      A(I)= 0.0
      B(I)= 0.0
      C(I)= 0.0
      D(I)= 0.0
      UNIT(I)= 0.0
      ZEROS(I)= 0.0
      GETPIX(I)= 0
10    CONTINUE

2     IF (FLAG.EQ.1) THEN
      FNAME= 'SPEC1'
      IC= 256 - (SIZE/4)
      IR= 20
      END IF

      IF (FLAG.EQ.2) THEN
      FNAME= 'SPEC2'
      IR= 100
      FLAG= FLAG + 1
      ENDIF

      CALL CHARZ (FNAME, 6)
      CALL PCKMOD (0)
      CALL RDISK (IER, FNAME, IC, IR, NCOL, NROW)
      CALL RDREC (IC, IR, NCOL, NROW, 0, GETPIX)
      K=1
      AVG= 0.0
      DO 13 I= ((SIZE/2)-(NCOL/2)), ((SIZE/2)+(NCOL/2)-1)
      IF (FLAG.EQ.1)      A(I)= REAL (GETPIX(K))
      IF (FLAG.EQ.3)      C(I)= REAL (GETPIX(K))
      UNIT(I)= 1.0
      AVG= GETPIX(K) + AVG
      K=K+1
13    CONTINUE
      AVG= AVG / (K-1)
C     COMPUTE SPECKLE STATISTICS
      STD= 0.0
      DO 15 I= ((SIZE/2)-(NCOL/2)), ((SIZE/2)+(NCOL/2)-1)
      STD= (GETPIX(I) - AVG)**2 + STD
15    CONTINUE
      STD= SQRT (STD / 255)
      WRITE (*, 41) AVG, STD
      CONTRS= STD / AVG
      WRITE (*, 42) CONTRS
41    FORMAT ( '/AVERAGE PIXEL INTENSITY IS ', F4.1/,
     & ' STANDARD DEVIATION IS ', F4.1)
42    FORMAT ( '/CONTRAST IS', F4.2)
      FLAG= FLAG + 1
      IF (FLAG.EQ.2) GOTO 2
```

```
C       COMPUTE IMAGE STATISTICS
        CALL PROBSTAT (A, SIZE, 300, 350)
        CALL PROBSTAT (C, SIZE, 450, 350)

C       PRODUCE A CORRELATION BY DIRECT EVALUATION
        CALL GRACORR (A, C, SIZE, TRIPEAK)
        PAUSE
        CALL FLREC (0, 150, 512, 362, 0)
        DO 18 I=1, SIZE
        A( I)= A( I) * ( -1)**I
        B( I)= B( I) * ( -1)**I
        C( I)= C( I) * ( -1)**I
        D( I)= D( I) * ( -1)**I
        UNIT( I)= UNIT( I) * ( -1)**I
        ZEROS( I)= ZEROS( I) * ( -1)**I
18      CONTINUE

        CALL FFT ( A, B, 1, SIZE, 1, -1)
        CALL FFT ( C, D, 1, SIZE, 1, -1)
        CALL FFT ( UNIT, ZEROS, 1, SIZE, 1, -1)

C       PRODUCE A PICTURE OF THE TRANSFORM
3       K= 1
        IMIN= TRIPEAK
        DO 17 I=1, SIZE
        IF (FLAG.EQ.4) THEN
        RHOLD= SQRT ( A(I)**2 + B(I)**2)
        ICFFT= 1
        IRFFT= 200
        ENDIF
        IF (FLAG.EQ.5) THEN
        RHOLD= SQRT ( C(I)**2 + D(I)**2)
        IRFFT= 220
        ENDIF
        TEMP= NINT (RHOLD)
        GETPIX (K)= TEMP
        IF (TEMP.LT.IMIN) IMIN= TEMP
        K= K+1
16      CONTINUE
17      CONTINUE
        K= K-1
        IMAX= 0
        DO 24 I=1, K
        GETPIX(I)= GETPIX(I) - IMIN
        IF (GETPIX(I).GT.IMAX) IMAX= GETPIX(I)
24      CONTINUE
        DO 26 I=1, K
        GETPIX(I)= GETPIX(I) * 254 / IMAX
26      CONTINUE
        CALL WRREC (ICFFT, IRFFT, SIZE, 1, 0, GETPIX)
        FLAG= FLAG + 1
        IF (FLAG.EQ.5) GOTO 3
```

```
      RETURN
      END
C     ******************************************
      SUBROUTINE XCORR (NCOL, NROW, TRIPEAK)
      INTEGER*2  ICCOR, IRCOR, PEAK, PEAKPOS
      INTEGER*2  GETPIX
      INTEGER*4  NROW, NCOL, SIZE
      INTEGER          TEMP
      REAL*4           MIN, MAX, RPIX
      REAL       A, B, C, D, UNIT, ZEROS
      REAL       TRIPEAK
      REAL*4           RHOLD,SHOLD, THOLD, UHOLD, VHOLD, WHOLD
      COMMON           A, B, C, D, UNIT, ZEROS
      COMMON           GETPIX
      DIMENSION  A(512), B(512), C(512), D(512)
      DIMENSION  UNIT(512), ZEROS(512)
      DIMENSION  GETPIX (512), RPIX (512)
      DATA       MIN /41943040./, SIZE /512/
C     (A-jB)(C+jD); RHOLD IS RE, SHOLD IS IM
      DO 14 I=1,SIZE
      RHOLD= A(I) * C(I) + B(I) * D(I)
      SHOLD= A(I) * D(I) - B(I) * C(I)
      THOLD= A(I) * UNIT(I) + B(I) * ZEROS(I)
      UHOLD= B(I) * UNIT(I) - A(I) * ZEROS(I)
      VHOLD= C(I) * UNIT(I) + D(I) * ZEROS(I)
      WHOLD= D(I) * UNIT(I) - C(I) * ZEROS(I)
      A(I)= RHOLD * ( -1)**I
      B(I)= SHOLD * ( -1)**I
      C(I)= THOLD * ( -1)**I
      D(I)= UHOLD  * ( -1)**I
      UNIT(I)= VHOLD * ( -1)**I
      ZEROS(I)= WHOLD * ( -1)**I

14    CONTINUE

      CALL FFT (A, B, 1, SIZE, 1, 1)

      CALL FFT (C, D, 1, SIZE, 1, 1)

      CALL FFT (UNIT, ZEROS, 1, SIZE, 1, 1)

C     COMPUTE MAGNITUDE VALUE OF CORRELATION FUNCTION
      DO 27 I=1, SIZE
      A(I)= SQRT (A(I)**2+B(I)**2)
27    CONTINUE
      N= SIZE
      DO 28 M=1, (SIZE/2)
      A(M)= A(M) / M
      A(N)= A(N) / M
      N= N - 1
28    CONTINUE
C     NORMALIZE CORRELATION FUNCTION
```

```
C       MU SQUARED

        DO 29 I= 1, SIZE
        C(I)= SQRT (C(I)**2+D(I)**2)
        UNIT(I)= SQRT (UNIT(I)**2+ZEROS(I)**2)
29      CONTINUE
C       DIVIDE FUNCTIONS C, UNIT BY TRIANGLE WITH PEAK= SIZE
        N= SIZE
        DO 26 M=1, (SIZE/2)
        C(M)= C(M) / M
        UNIT(M)= UNIT(M) / M
        C(N)= C(N) / M
        UNIT(N)= UNIT(N) / M
        N= N-1
26      CONTINUE
C       A IS MU SQUARED, ABS VALUE
        DO 30 I=1, SIZE
        A(I)= (A(I) / (C(I) * UNIT(I))) - 1
        IF (A(I).LT.0) A(I)= 0.0
        IF ((I.LT.128).OR.(I.GT.384)) A(I)= 0.0
30      CONTINUE
        K=1
        DO 17 I= 1, SIZE
C       FIND MAX AMPLITUDE
        RPIX(K)= A(I)
        IF (RPIX(K).LT.MIN) THEN
        MIN= RPIX(K)
        END IF
        K= K+1
17      CONTINUE
        MAX= 0
        DO 19 I=1, SIZE
        RPIX(I)= RPIX(I) - MIN
        IF (RPIX(I).GT.MAX) THEN
        MAX= RPIX(I)
        END IF
19      CONTINUE
        DO 20 I=1, SIZE
        GETPIX(I)= NINT( RPIX(I) * 254 / MAX)
20      CONTINUE
        ICCOR= 1
        IRCOR= 200
        PEAK= 0.0
        DO 21 I=1, SIZE
        IF (GETPIX(I).GT.PEAK) THEN
        PEAK= GETPIX(I)
        PEAKPOS=I
        ENDIF
21      CONTINUE
        PAUSE
        WRITE (*, 80) PEAKPOS, PEAKPOS-256
80      FORMAT ('\POSITION OF PEAK IS AT: ', I3,
```

```
     & '   TRANSLATION OF', I4, ' PIXELS')
        CALL FLREC (0, 150, 512, 362, 0)
        CALL WRREC (ICCOR, IRCOR, SIZE, 1, 0, GETPIX)
        CALL QLOOK (GETPIX, SIZE)
C       WRITE TICK MARKS ON SCREEN
        DO 22 J=1, 64
        L= J*8
        CALL WRPXL (L, 205, 100)
22      CONTINUE
        CALL WRPXL (256, 210, 155)
        CALL WRPXL (PEAKPOS, 210, 255)
        RETURN
        END
C       ******************************************************
C       ROUTINE PERFORMS GRAPHICAL CORRELATION
        SUBROUTINE GRACORR (AR1, AR2, NUMBER, TRIPEAK)
        REAL                AR1, AR2, PRODUCT, OUTPUT, SUM
        REAL          RMIN, RMAX, TRIPEAK, PARTIAL
        INTEGER*2     HETPIX, RPEAK, RPEAKPOS
        INTEGER*4     NUMBER
        DIMENSION     AR1(512), AR2(512), PRODUCT(512), OUTPUT(512)
        DIMENSION     HETPIX(512)
        DO 61 I= 1, NUMBER
        SUM= 0.0
        DO 60 J= 1, NUMBER
        K= J + I - (NUMBER/2)
        IF ((K.LT.1).OR.(K.GT.NUMBER)) PRODUCT(J)= 0.0
        IF ((K.GE.1).AND.(K.LE.NUMBER)) PRODUCT(J)= AR1(J) * AR2(K)
        SUM= SUM + PRODUCT (J)
60      CONTINUE
        OUTPUT(I)= SUM
61      CONTINUE

C       COMPUTE PEAK VALUE OF TRIANGLE
        TRIPEAK= 0
        DO 62 N=1, NUMBER
        PARTIAL= AR1(N) * AR2(N)
        TRIPEAK= TRIPEAK + PARTIAL
62      CONTINUE

C
C       DIVIDE BY TRIANGLE
        N= NUMBER
        DO 66 M=1, (NUMBER/2)
        OUTPUT(M)= OUTPUT(M) / M
        OUTPUT(N)= OUTPUT(N) / M
        IF (OUTPUT(M).LT.0) OUTPUT(M)= 0
        IF (OUTPUT(N).LT.0) OUTPUT(N)= 0
        N= N - 1
66      CONTINUE
        DO 67 I=1, 127
        OUTPUT(I)= 0.0
```

```
67      CONTINUE
        DO 68 I= 385, 512
        OUTPUT(I)= 0.0
68      CONTINUE

        K=1
        RMIN= TRIPEAK
        DO 70 I=1, NUMBER
        PRODUCT(K)= OUTPUT(I)
        IF (PRODUCT(K).LT.RMIN) THEN
        RMIN= PRODUCT(K)
        ENDIF
        K=K+1
70      CONTINUE
        K=K-1
        RMAX= 0.0
        DO 72 I=1, K
        PRODUCT(I)= PRODUCT(I) - RMIN
        IF (PRODUCT(I).GT.RMAX) RMAX= PRODUCT(I)
72      CONTINUE
        RPEAK= 0.0
        DO 74 I=1, K
        HETPIX(I)= NINT (PRODUCT(I) * 250 / RMAX)
        IF (HETPIX(I).GT.RPEAK) THEN
        RPEAK= HETPIX(I)
        RPEAKPOS= I
        ENDIF
74      CONTINUE
        PAUSE
        WRITE (*, 80) RPEAKPOS, RPEAKPOS-256
80      FORMAT ('\POSITION OF PEAK IS AT: ', I3,
     &  '   TRANSLATION OF ', I4, ' PIXELS')
        CALL FLREC (0, 150, 512, 362, 0)
        CALL WRREC (1, 200, NUMBER, 1, 0, HETPIX)
C       DRAW MEASURE STICK
        DO 22 J=1, 64
        L= J*8
        CALL WRPXL (L, 205, 100)
22      CONTINUE
        CALL WRPXL (256, 210, 155)
        CALL WRPXL (RPEAKPOS, 210, 255)
        CALL QLOOK (HETPIX, NUMBER)
        RETURN
        END
C       ***************************************************
        SUBROUTINE QLOOK (ARRAY, NUMBER)
        INTEGER*2   ARRAY, X, Y, VALUE
        INTEGER*4   NUMBER
        DIMENSION   ARRAY(512)
        DATA        VALUE/175/
        DO 15 I=1, NUMBER
        X= I
```

```
      Y= 470 - (ARRAY(I)/2)
      CALL WRPXL (X, Y, VALUE)
15    CONTINUE
      RETURN
      END
C     ******************************************************
      SUBROUTINE PROBSTAT (AR, NUMBER, VERPOS, HORPOS)
      REAL        AR, MAX, AVG
      REAL        INTENT, PROB
      INTEGER*2   SUM, J, VERPOS, HORPOS, FLAG
      INTEGER*4   NUMBER
      DIMENSION   AR(512), INTENT(128), PROB(128)
      DATA        PROB/128 * 0/
      SUM= 0
      MAX= 0
      DO 10 I= 129, 382
      SUM= AR(I) + SUM
      IF (AR(I).GT.MAX) MAX= AR(I)
10    CONTINUE
      AVG= SUM * 2 / NUMBER
      FLAG= 0
      DO 13 I= 1, 128
      INTENT(I)= 256 * I / 128
      PROB(I)= 0
      IF (FLAG.GT.0) GOTO 13
      IF (INTENT(I).GE.AVG) FLAG= I
13    CONTINUE
C     MARK THE POSITION OF AVERAGE INTENSITY
      CALL WRPXL ( (HORPOS+FLAG), (VERPOS+5), 200)
      DO 17 I= 129, 384
      DO 16 J= 1, 128
      IF (AR(I).LT.INTENT(J+1)) THEN
      PROB(J)= PROB(J) + 1
      GOTO 17
      ENDIF
16    CONTINUE
17    CONTINUE
      CALL STATLOOK (INTENT, PROB, NUMBER, VERPOS, HORPOS)
      RETURN
      END
C     ******************************************************
      SUBROUTINE STATLOOK (AR1, AR2, NUMBER, VERPOS, HORPOS)
      REAL        AR1, AR2
      INTEGER*4   NUMBER
      INTEGER*2   VERPOS, HORPOS, X, Y, VALUE
      DIMENSION   AR1(128), AR2(128)
      DATA        VALUE/145/
      DO 10 I=1, 128
      X= HORPOS + I
      Y= VERPOS - (AR2(I)*2)
      CALL WRPXL (X, Y, VALUE)
10    CONTINUE
```

```
      RETURN
      END
C     ************************************************************
      SUBROUTINE AREAPR
      INTEGER*2  STATPX, NUMPIX
      INTEGER*2  X, Y
      DIMENSION  STATPX (16384), NUMPIX(256)
      DO 8 I= 1, 256
      NUMPIX(I)= 0
8     CONTINUE
      CALL RDREC (192, 192, 128, 128, 0, STATPX)
      DO 9 I=1, 16384
      NUMPIX (STATPX(I))= NUMPIX (STATPX(I)) + 1
9     CONTINUE
      DO 10 I=1, 256
      X= I + 128
      Y= 470 - (NUMPIX(I) / 8)
      CALL WRPXL (X, Y, 250)
10    CONTINUE
      RETURN
      END
C     ************************************************************
   SUBROUTINE FFT( A, B, NSEG, N, NSPN, ISN)
C
C     THIS IS TAKEN FROM:
C     'PROGRAMS FOR DIGITAL SIGNAL PROCESSING'
C      EDITED BY THE DIGITAL SIGNAL PROCESSING COMMITTEE;
C      IEEE ACOUSTICS , SPEECH, AND SIGNAL PROCESSING SOCIETY.
C      IEEE PRESS, 1979
C
C     ARRAYS A AND B ORIGINALLY HOLD THE REAL AND IMAGINARY
C   COMPONENTS OF THE DATA, AND RETURN THE REAL AND
C         IMAGINARY COMPONENTS OF THE RESULTING FOURIER
C     COEFFICIENTS.
C      MULTIVARIATE DATA IS INDEXED ACCORDING TO THE FORTRAN
C   ARRAY ELEMENT SUCCESSOR FUNCTION, WITHOUT LIMIT
C   ON THE NUMBER OF IMPLIED MULTIPLE SUBSCRIPTS.
C   THE SUBROUTINE IS CALLED ONCE FOR EACH VARIATE.
C     THE CALLS FOR A MULTIVARIATE TRANSFORM MAY BE IN ANY
C     ORDER.
C
C     N IS THE DIMENSION OF THE CURRENT VARIABLE.
C     NSPN IS THE SPACING OF CONSECUTIVE DATA VALUES
C   WHILE INDEXING THE CURRENT VARIABLE.
C     NSEG*N*NSPN IS THE TOTAL NUMBER OF COMPLEX DATA VALUES.
C     THE SIGN OF ISN DETERMINES THE SIGN OF THE COMPLEX
C   EXPONENTIAL, AND THE MAGNITUDE OF ISN IS NORMALLY ONE.
C    THE MAGNITUDE OF ISN DETERMINES THE INDEXING INCREMENT
C     FOR A&B.
C
C     IF FFT IS CALLED TWICE, WITH OPPOSITE SIGNS ON ISN, AN
C     IDENTITY TRANSFORMATION IS DONE...CALLS CAN BE IN EITHER
```

```
C       ORDER.
C     THE RESULTS ARE SCALED BY 1/N WHEN THE SIGN OF ISN IS POSITIVE.
C
C       A TRI-VARIATE TRANSFORM WITH A(N1,N2,N3), B(N1,N2,N3)
C       IS COMPUTED BY
C       CALL FFT(A,B,N2*N3,N1,1,-1)
C       CALL FFT(A,B,N3,N2,N1,-1)
C       CALL FFT(A,B,1,N3,N1*N2,-1)
C
C       A SINGLE-VARIATE TRANSFORM OF N COMPLEX DATA VALUES IS
C       COMPUTED BY
C       CALL FFT(A,B,1,N,1,-1)
C
C       THE DATA MAY ALTERNATIVELY BE STORED IN A SINGLE COMPLEX
C     ARRAY A, THEN THE MAGNITUDE OF ISN CHANGED TO TWO TO
C     GIVE THE CORRECT INDEXING INCREMENT AND A(2) USED TO
C     PASS THE INITIAL ADDRESS FOR THE SEQUENCE OF IMAGINARY
C     VALUES, E.G.
C       CALL FFT(A,A(2),NSEG,N,NSPN,-2)
C
C       ARRAY NFAC IS WORKING STORAGE FOR FACTORING N.  THE
C       SMALLEST
C     NUMBER EXCEEDING THE 15 LOCATIONS PROVIDED IS 12,754,584.
C
C       DOUBLE PRECISION DSTACK(2500)   :   THESE LINES MAY BE
C       NECESSARY
C       COMMON /CSTAK/ DSTACK           : IN THE CALLING ROUTINE.
C
      DIMENSION A( 1), B( 1), NFAC( 15)
CC
CC
      COMMON/ CSTAK/ DSTAK( 2500)
      DOUBLE PRECISION DSTAK
      INTEGER ISTAK( 5000)
      REAL RSTAK( 5000)
C
      INTEGER ISIZE( 5)
      EQUIVALENCE( ISTAK( 1), LOUT)
      EQUIVALENCE( ISTAK( 2), LNOW)
      EQUIVALENCE( ISTAK( 3), LUSED)
      EQUIVALENCE( ISTAK( 4), LMAX)
      EQUIVALENCE( ISTAK( 5), LBOOK)
      EQUIVALENCE( ISTAK( 6), ISIZE( 1))
      EQUIVALENCE( DSTAK( 1), ISTAK( 1))
      EQUIVALENCE( DSTAK( 1), RSTAK( 1))
C
C DETERMINE THE FACTORS OF N
C
      LOGICAL* 2 FIRST
      DATA FIRST/ .TRUE./
C     DATA ISIZE/ 1, 1, 1, 2, 2/
C     DATA LOUT, LNOW, LUSED, LMAX, LBOOK/ 0, 10, 10, 5000, 10/
```

```
C INIT DATA IN COMMON.
      IF( FIRST) THEN
         ISIZE( 1)= 1
         ISIZE( 2)= 1
         ISIZE( 3)= 1
         ISIZE( 4)= 2
         ISIZE( 5)= 2
         LOUT= 0
         LNOW= 10
         LUSED= 10
         LMAX= 5000
         LBOOK= 10
         FIRST= .FALSE.
      END IF
      M= 0
         write (*, *) LOUT
      NF= IABS( N)
      K= NF
      IF( NF .EQ. 1) RETURN
      NSPAN= IABS( NF* NSPN)
      NTOT= IABS( NSPAN* NSEG)
      IF( ISN* NTOT .NE. 0) GO TO 20
      IERR= 6
         write (*, *) LOUT
      WRITE( *, 9999) NSEG, N, NSPN, ISN
9999  FORMAT( 31H ERROR - ZERO IN FFT PARAMETERS, 4I10)
      RETURN
C
  10  M= M+ 1
      NFAC( M)= 4
      K= K/ 16
  20  IF( K-( K/ 16)* 16 .EQ. 0) GO TO 10
      J= 3
      JJ= 9
      GO TO 40
  30  M= M+ 1
      NFAC( M)= J
      K= K/ JJ
  40  IF( MOD( K, JJ) .EQ. 0) GO TO 30
      J= J+ 2
      JJ= J** 2
      IF( JJ .LE. K) GO TO 40
      IF( K .GT. 4) GO TO 50
      KT= M
      NFAC( M+ 1)= K
      IF( K .NE. 1) M= M+ 1
      GO TO 90
  50  IF( K-( K/ 4)* 4 .NE. 0) GO TO 60
      M= M+ 1
      NFAC( M)= 2
      K= K/ 4
C ALL SQUARE FACTORS OUT NOW, BUT K .GE. 5 STILL
```

```
 60  KT= M
     MAXP= MAX0( KT+ KT+ 2, K- 1)
     J= 2
 70  IF( MOD( K, J) .NE. 0) GO TO 80
     M= M+ 1
     NFAC( M)= J
     K= K/ J
 80  J=(( J+ 1)/ 2)* 2+ 1
     IF( J .LE. K) GO TO 70
 90  IF( M .LE. KT+ 1) MAXP= M+ KT+ 1
     IF( M+ KT .GT. 15) GO TO 120
     IF( KT .EQ. 0) GO TO 110
     J= KT
100  M= M+ 1
     NFAC( M)= NFAC( J)
     J= J- 1
     IF( J .NE. 0) GO TO 100
C
110  MAXF= M- KT
     MAXF= NFAC( MAXF)
     IF( KT .GT. 0) MAXF= MAX0( NFAC( KT), MAXF)
     J= ISTKGT( MAXF* 4, 3)
     JJ= J+ MAXF
     J2= JJ+ MAXF
     J3= J2+ MAXF
     K= ISTKGT( MAXP, 2)
        write (*, *) LOUT
     CALL FFTMX( A, B, NTOT, NF, NSPAN, ISN, M, KT, RSTAK( J),
    1 RSTAK( JJ), RSTAK( J2), RSTAK( J3), ISTAK( K), NFAC)
     CALL ISTKRL( 2)
     RETURN
C
120  IERR= 6
     WRITE( *, 9998) N
9998  FORMAT( 50H ERROR - FFT PARAMETER N HAS MORE THAN 15
C       FACTORS-,
    1 I20)
     RETURN
     END
C
C-------------------------------------------------------------------------
C SUBROUTINE:  FFTMX
C CALLED BY SUBROUTINE 'FFT' TO COMPUTE MIXED-RADIX FOURIER
C       TRANSFORM
C-------------------------------------------------------------------------
C
     SUBROUTINE FFTMX( A, B, NTOT, N, NSPAN, ISN, M, KT, AT, CK,
    1 BT, SK, NP, NFAC)
C
     DIMENSION A( N), B( N), AT( N), CK( N), BT( N), SK( N), NP( N),
    1 NFAC( N)
C
```

```
      write (*, *) N
    INC= IABS( ISN)
    NT= INC* NTOT
    KS= INC* NSPAN
    RAD= ATAN( 1.0)
    S72= RAD/ 0.625
    C72= COS( S72)
    S72= SIN( S72)
    S120= SQRT( 0.75)
    IF( ISN .GT. 0) GO TO 10
    S72=- S72
    S120=- S120
    RAD=- RAD
    GO TO 30
C
C SCALE BY 1/N FOR ISN .GT. 0
C
 10 AK= 1.0/ FLOAT( N)
    DO 20 J= 1, NT, INC
    A( J)= A( J)* AK
    B( J)= B( J)* AK
 20 CONTINUE
C
 30 KSPAN= KS
    NN= NT- INC
    JC= KS/ N
C
C SIN, COS VALUES ARE RE-INITIALIZED EACH LIM STEPS
C
    LIM= 32
    KLIM= LIM* JC
    I= 0
    JF= 0
    MAXF= M- KT
    MAXF= NFAC( MAXF)
    IF( KT .GT. 0) MAXF= MAX0( NFAC( KT), MAXF)
C
C COMPUTE FOURIER TRANSFORM
C
 40 DR= 8.0* FLOAT( JC)/ FLOAT( KSPAN)
    CD= 2.0* SIN( 0.5* DR* RAD)** 2
    SD= SIN( DR* RAD)
    KK= 1
    I= I+ 1
    IF( NFAC( I) .NE. 2) GO TO 110
C
C TRANSFORM FOR FACTOR OF 2 (INCLUDING ROTATION FACTOR)
C
    KSPAN= KSPAN/ 2
    K1= KSPAN+ 2
 50 K2= KK+ KSPAN
    AK= A( K2)
```

```
      BK= B( K2)
      A( K2)= A( KK)- AK
      B( K2)= B( KK)- BK
      A( KK)= A( KK)+ AK
      B( KK)= B( KK)+ BK
      KK= K2+ KSPAN
      IF( KK .LE. NN) GO TO 50
      KK= KK- NN
      IF( KK .LE. JC) GO TO 50
      IF( KK .GT. KSPAN) GO TO 350
   60 C1= 1.0- CD
      S1= SD
      MM= MIN0( K1/ 2, KLIM)
      GO TO 80
   70 AK= C1-( CD* C1+ SD* S1)
      S1=( SD* C1- CD* S1)+ S1
C
C THE FOLLOWING THREE STATEMENTS COMPENSATE FOR TRUNCATION
C ERROR.  IF ROUNDED ARITHMETIC IS USED, SUBSTITUTE
C C1=AK
C
      C1= 0.5/( AK** 2+ S1** 2)+ 0.5
      S1= C1* S1
      C1= C1* AK
   80 K2= KK+ KSPAN
      AK= A( KK)- A( K2)
      BK= B( KK)- B( K2)
      A( KK)= A( KK)+ A( K2)
      B( KK)= B( KK)+ B( K2)
      A( K2)= C1* AK- S1* BK
      B( K2)= S1* AK+ C1* BK
      KK= K2+ KSPAN
      IF( KK .LT. NT) GO TO 80
      K2= KK- NT
      C1=- C1
      KK= K1- K2
      IF( KK .GT. K2) GO TO 80
      KK= KK+ JC
      IF( KK .LE. MM) GO TO 70
      IF( KK .LT. K2) GO TO 90
      K1= K1+ INC+ INC
      KK=( K1- KSPAN)/ 2+ JC
      IF( KK .LE. JC+ JC) GO TO 60
      GO TO 40
   90 S1= FLOAT(( KK- 1)/ JC)* DR* RAD
      C1= COS( S1)
      S1= SIN( S1)
      MM= MIN0( K1/ 2, MM+ KLIM)
      GO TO 80
C
C TRANSFORM FOR FACTOR OF 3 (OPTIONAL CODE)
C
```

```
100  K1= KK+ KSPAN
     K2= K1+ KSPAN
     AK= A( KK)
     BK= B( KK)
     AJ= A( K1)+ A( K2)
     BJ= B( K1)+ B( K2)
     A( KK)= AK+ AJ
     B( KK)= BK+ BJ
     AK=- 0.5* AJ+ AK
     BK=- 0.5* BJ+ BK
     AJ=( A( K1)- A( K2))* S120
     BJ=( B( K1)- B( K2))* S120
     A( K1)= AK- BJ
     B( K1)= BK+ AJ
     A( K2)= AK+ BJ
     B( K2)= BK- AJ
     KK= K2+ KSPAN
     IF( KK .LT. NN) GO TO 100
     KK= KK- NN
     IF( KK .LE. KSPAN) GO TO 100
     GO TO 290
C
C TRANSFORM FOR FACTOR OF 4
C
110  IF( NFAC( I) .NE. 4) GO TO 230
     KSPNN= KSPAN
     KSPAN= KSPAN/ 4
120  C1= 1.0
     S1= 0
     MM= MIN0( KSPAN, KLIM)
     GO TO 150
130  C2= C1-( CD* C1+ SD* S1)
     S1=( SD* C1- CD* S1)+ S1
C
C THE FOLLOWING THREE STATEMENTS COMPENSATE FOR TRUNCATION
C ERROR.  IF ROUNDED ARITHMETIC IS USED, SUBSTITUTE
C C1=C2
C
     C1= 0.5/( C2** 2+ S1** 2)+ 0.5
     S1= C1* S1
     C1= C1* C2
140  C2= C1** 2- S1** 2
     S2= C1* S1* 2.0
     C3= C2* C1- S2* S1
     S3= C2* S1+ S2* C1
150  K1= KK+ KSPAN
     K2= K1+ KSPAN
     K3= K2+ KSPAN
     AKP= A( KK)+ A( K2)
     AKM= A( KK)- A( K2)
     AJP= A( K1)+ A( K3)
     AJM= A( K1)- A( K3)
```

```
      A( KK)= AKP+ AJP
      AJP= AKP- AJP
      BKP= B( KK)+ B( K2)
      BKM= B( KK)- B( K2)
      BJP= B( K1)+ B( K3)
      BJM= B( K1)- B( K3)
      B( KK)= BKP+ BJP
      BJP= BKP- BJP
      IF( ISN .LT. 0) GO TO 180
      AKP= AKM- BJM
      AKM= AKM+ BJM
      BKP= BKM+ AJM
      BKM= BKM- AJM
      IF( S1 .EQ. 0.0) GO TO 190
  160 A( K1)= AKP* C1- BKP* S1
      B( K1)= AKP* S1+ BKP* C1
      A( K2)= AJP* C2- BJP* S2
      B( K2)= AJP* S2+ BJP* C2
      A( K3)= AKM* C3- BKM* S3
      B( K3)= AKM* S3+ BKM* C3
      KK= K3+ KSPAN
      IF( KK .LE. NT) GO TO 150
  170 KK= KK- NT+ JC
      IF( KK .LE. MM) GO TO 130
      IF( KK .LT. KSPAN) GO TO 200
      KK= KK- KSPAN+ INC
      IF( KK .LE. JC) GO TO 120
      IF( KSPAN .EQ. JC) GO TO 350
      GO TO 40
  180 AKP= AKM+ BJM
      AKM= AKM- BJM
      BKP= BKM- AJM
      BKM= BKM+ AJM
      IF( S1 .NE. 0.0) GO TO 160
  190 A( K1)= AKP
      B( K1)= BKP
      A( K2)= AJP
      B( K2)= BJP
      A( K3)= AKM
      B( K3)= BKM
      KK= K3+ KSPAN
      IF( KK .LE. NT) GO TO 150
      GO TO 170
  200 S1= FLOAT(( KK- 1)/ JC)* DR* RAD
      C1= COS( S1)
      S1= SIN( S1)
      MM= MIN0( KSPAN, MM+ KLIM)
      GO TO 140
C
C TRANSFORM FOR FACTOR OF 5 (OPTIONAL CODE)
C
  210 C2= C72** 2- S72** 2
```

```
      S2= 2.0* C72* S72
  220 K1= KK+ KSPAN
      K2= K1+ KSPAN
      K3= K2+ KSPAN
      K4= K3+ KSPAN
      AKP= A( K1)+ A( K4)
      AKM= A( K1)- A( K4)
      BKP= B( K1)+ B( K4)
      BKM= B( K1)- B( K4)
      AJP= A( K2)+ A( K3)
      AJM= A( K2)- A( K3)
      BJP= B( K2)+ B( K3)
      BJM= B( K2)- B( K3)
      AA= A( KK)
      BB= B( KK)
      A( KK)= AA+ AKP+ AJP
      B( KK)= BB+ BKP+ BJP
      AK= AKP* C72+ AJP* C2+ AA
      BK= BKP* C72+ BJP* C2+ BB
      AJ= AKM* S72+ AJM* S2
      BJ= BKM* S72+ BJM* S2
      A( K1)= AK- BJ
      A( K4)= AK+ BJ
      B( K1)= BK+ AJ
      B( K4)= BK- AJ
      AK= AKP* C2+ AJP* C72+ AA
      BK= BKP* C2+ BJP* C72+ BB
      AJ= AKM* S2- AJM* S72
      BJ= BKM* S2- BJM* S72
      A( K2)= AK- BJ
      A( K3)= AK+ BJ
      B( K2)= BK+ AJ
      B( K3)= BK- AJ
      KK= K4+ KSPAN
      IF( KK .LT. NN) GO TO 220
      KK= KK- NN
      IF( KK .LE. KSPAN) GO TO 220
      GO TO 290
C
C TRANSFORM FOR ODD FACTORS
C
  230 K= NFAC( I)
      KSPNN= KSPAN
      KSPAN= KSPAN/ K
      IF( K .EQ. 3) GO TO 100
      IF( K .EQ. 5) GO TO 210
      IF( K .EQ. JF) GO TO 250
      JF= K
      S1= RAD/( FLOAT( K)/ 8.0)
      C1= COS( S1)
      S1= SIN( S1)
      CK( JF)= 1.0
```

```
      SK( JF)= 0.0
      J= 1
240   CK( J)= CK( K)* C1+ SK( K)* S1
      SK( J)= CK( K)* S1- SK( K)* C1
      K= K- 1
      CK( K)= CK( J)
      SK( K)=- SK( J)
      J= J+ 1
      IF( J .LT. K) GO TO 240
250   K1= KK
      K2= KK+ KSPNN
      AA= A( KK)
      BB= B( KK)
      AK= AA
      BK= BB
      J= 1
      K1= K1+ KSPAN
260   K2= K2- KSPAN
      J= J+ 1
      AT( J)= A( K1)+ A( K2)
      AK= AT( J)+ AK
      BT( J)= B( K1)+ B( K2)
      BK= BT( J)+ BK
      J= J+ 1
      AT( J)= A( K1)- A( K2)
      BT( J)= B( K1)- B( K2)
      K1= K1+ KSPAN
      IF( K1 .LT. K2) GO TO 260
      A( KK)= AK
      B( KK)= BK
      K1= KK
      K2= KK+ KSPNN
      J= 1
270   K1= K1+ KSPAN
      K2= K2- KSPAN
      JJ= J
      AK= AA
      BK= BB
      AJ= 0.0
      BJ= 0.0
      K= 1
280   K= K+ 1
      AK= AT( K)* CK( JJ)+ AK
      BK= BT( K)* CK( JJ)+ BK
      K= K+ 1
      AJ= AT( K)* SK( JJ)+ AJ
      BJ= BT( K)* SK( JJ)+ BJ
      JJ= JJ+ J
      IF( JJ .GT. JF) JJ= JJ- JF
      IF( K .LT. JF) GO TO 280
      K= JF- J
      A( K1)= AK- BJ
```

```
      B( K1)= BK+ AJ
      A( K2)= AK+ BJ
      B( K2)= BK- AJ
      J= J+ 1
      IF( J .LT. K) GO TO 270
      KK= KK+ KSPNN
      IF( KK .LE. NN) GO TO 250
      KK= KK- NN
      IF( KK .LE. KSPAN) GO TO 250
C
C MULTIPLY BY ROTATION FACTOR (EXCEPT FOR FACTORS OF 2 AND 4)
C
  290 IF( I .EQ. M) GO TO 350
      KK= JC+ 1
  300 C2= 1.0- CD
      S1= SD
      MM= MIN0( KSPAN, KLIM)
      GO TO 320
  310 C2= C1-( CD* C1+ SD* S1)
      S1= S1+( SD* C1- CD* S1)
C
C THE FOLLOWING THREE STATEMENTS COMPENSATE FOR TRUNCATION
C ERROR.  IF ROUNDED ARITHMETIC IS USED, THEY MAY
C BE DELETED.
C
      C1= 0.5/( C2** 2+ S1** 2)+ 0.5
      S1= C1* S1
      C2= C1* C2
  320 C1= C2
      S2= S1
      KK= KK+ KSPAN
  330 AK= A( KK)
      A( KK)= C2* AK- S2* B( KK)
      B( KK)= S2* AK+ C2* B( KK)
      KK= KK+ KSPNN
      IF( KK .LE. NT) GO TO 330
      AK= S1* S2
      S2= S1* C2+ C1* S2
      C2= C1* C2- AK
      KK= KK- NT+ KSPAN
      IF( KK .LE. KSPNN) GO TO 330
      KK= KK- KSPNN+ JC
      IF( KK .LE. MM) GO TO 310
      IF( KK .LT. KSPAN) GO TO 340
      KK= KK- KSPAN+ JC+ INC
      IF( KK .LE. JC+ JC) GO TO 300
      GO TO 40
  340 S1= FLOAT(( KK- 1)/ JC)* DR* RAD
      C2= COS( S1)
      S1= SIN( S1)
      MM= MIN0( KSPAN, MM+ KLIM)
      GO TO 320
```

```
C
C PERMUTE THE RESULTS TO NORMAL ORDER---DONE IN TWO STAGES
C PERMUTATION FOR SQUARE FACTORS OF N
C
 350  NP( 1)= KS
      IF( KT .EQ. 0) GO TO 440
      K= KT+ KT+ 1
      IF( M .LT. K) K= K- 1
      J= 1
      NP( K+ 1)= JC
 360  NP( J+ 1)= NP( J)/ NFAC( J)
      NP( K)= NP( K+ 1)* NFAC( J)
      J= J+ 1
      K= K- 1
      IF( J .LT. K) GO TO 360
      K3= NP( K+ 1)
      KSPAN= NP( 2)
      KK= JC+ 1
      K2= KSPAN+ 1
      J= 1
      IF( N .NE. NTOT) GO TO 400
C
C PERMUTATION FOR SINGLE-VARIATE TRANSFORM (OPTIONAL CODE)
C
 370  AK= A( KK)
      A( KK)= A( K2)
      A( K2)= AK
      BK= B( KK)
      B( KK)= B( K2)
      B( K2)= BK
      KK= KK+ INC
      K2= KSPAN+ K2
      IF( K2 .LT. KS) GO TO 370
 380  K2= K2- NP( J)
      J= J+ 1
      K2= NP( J+ 1)+ K2
      IF( K2 .GT. NP( J)) GO TO 380
      J= 1
 390  IF( KK .LT. K2) GO TO 370
      KK= KK+ INC
      K2= KSPAN+ K2
      IF( K2 .LT. KS) GO TO 390
      IF( KK .LT. KS) GO TO 380
      JC= K3
      GO TO 440
C
C PERMUTATION FOR MULTIVARIATE TRANSFORM
C
 400  K= KK+ JC
 410  AK= A( KK)
      A( KK)= A( K2)
      A( K2)= AK
```

```
      BK= B( KK)
      B( KK)= B( K2)
      B( K2)= BK
      KK= KK+ INC
      K2= K2+ INC
      IF( KK .LT. K) GO TO 410
      KK= KK+ KS- JC
      K2= K2+ KS- JC
      IF( KK .LT. NT) GO TO 400
      K2= K2- NT+ KSPAN
      KK= KK- NT+ JC
      IF( K2 .LT. KS) GO TO 400
  420 K2= K2- NP( J)
      J= J+ 1
      K2= NP( J+ 1)+ K2
      IF( K2 .GT. NP( J)) GO TO 420
      J= 1
  430 IF( KK .LT. K2) GO TO 400
      KK= KK+ JC
      K2= KSPAN+ K2
      IF( K2 .LT. KS) GO TO 430
      IF( KK .LT. KS) GO TO 420
      JC= K3
  440 IF( 2* KT+ 1 .GE. M) RETURN
      KSPNN= NP( KT+ 1)
C
C PERMUTATION FOR SQUARE-FREE FACTORS OF N
C
      J= M- KT
      NFAC( J+ 1)= 1
  450 NFAC( J)= NFAC( J)* NFAC( J+ 1)
      J= J- 1
      IF( J .NE. KT) GO TO 450
      KT= KT+ 1
      NN= NFAC( KT)- 1
      JJ= 0
      J= 0
      GO TO 480
  460 JJ= JJ- K2
      K2= KK
      K= K+ 1
      KK= NFAC( K)
  470 JJ= KK+ JJ
      IF( JJ .GE. K2) GO TO 460
      NP( J)= JJ
  480 K2= NFAC( KT)
      K= KT+ 1
      KK= NFAC( K)
      J= J+ 1
      IF( J .LE. NN) GO TO 470
C
C DETERMINE THE PERMUTATION CYCLES OF LENGTH GREATER THAN 1
```

```
C
      J= 0
      GO TO 500
  490 K= KK
      KK= NP( K)
      NP( K)=- KK
      IF( KK .NE. J) GO TO 490
      K3= KK
  500 J= J+ 1
      KK= NP( J)
      IF( KK .LT. 0) GO TO 500
      IF( KK .NE. J) GO TO 490
      NP( J)=- J
      IF( J .NE. NN) GO TO 500
      MAXF= INC* MAXF
C
C REORDER A AND B, FOLLOWING THE PERMUTATION CYCLES
C
      GO TO 570
  510 J= J- 1
      IF( NP( J) .LT. 0) GO TO 510
      JJ= JC
  520 KSPAN= JJ
      IF( JJ .GT. MAXF) KSPAN= MAXF
      JJ= JJ- KSPAN
      K= NP( J)
      KK= JC* K+ I+ JJ
      K1= KK+ KSPAN
      K2= 0
  530 K2= K2+ 1
      AT( K2)= A( K1)
      BT( K2)= B( K1)
      K1= K1- INC
      IF( K1 .NE. KK) GO TO 530
  540 K1= KK+ KSPAN
      K2= K1- JC*( K+ NP( K))
      K=- NP( K)
  550 A( K1)= A( K2)
      B( K1)= B( K2)
      K1= K1- INC
      K2= K2- INC
      IF( K1 .NE. KK) GO TO 550
      KK= K2
      IF( K .NE. J) GO TO 540
      K1= KK+ KSPAN
      K2= 0
  560 K2= K2+ 1
      A( K1)= AT( K2)
      B( K1)= BT( K2)
      K1= K1- INC
      IF( K1 .NE. KK) GO TO 560
      IF( JJ .NE. 0) GO TO 520
```

```
      IF( J .NE. 1) GO TO 510
  570 J= K3+ 1
      NT= NT- KSPNN
      I= NT- INC+ 1
      IF( NT .GE. 0) GO TO 510
      RETURN
      END
C
C-----------------------------------------------------------------------
C FUNCTION:  ISTKGT(NITEMS,ITYPE)
C ALLOCATES WORKING STORAGE FOR NITEMS OF ITYPE, AS FOLLOWS
C
C 1 - LOGICAL
C 2 - INTEGER
C 3 - REAL
C 4 - DOUBLE PRECISION
C 5 - COMPLEX
C
C-------------------------------------------- ----------------------
C
      INTEGER FUNCTION ISTKGT( NITEMS, ITYPE)
C
      COMMON/ CSTAK/ DSTAK( 2500)
C
      DOUBLE PRECISION DSTAK
      INTEGER ISTAK( 5000)
      INTEGER ISIZE( 5)
C
      EQUIVALENCE( DSTAK( 1), ISTAK( 1))
      EQUIVALENCE( ISTAK( 1), LOUT)
      EQUIVALENCE( ISTAK( 2), LNOW)
      EQUIVALENCE( ISTAK( 3), LUSED)
      EQUIVALENCE( ISTAK( 4), LMAX)
      EQUIVALENCE( ISTAK( 5), LBOOK)
      EQUIVALENCE( ISTAK( 6), ISIZE( 1))
C
      ISTKGT=( LNOW* ISIZE( 2)- 1)/ ISIZE( ITYPE)+ 2
      I=(( ISTKGT- 1+ NITEMS)* ISIZE( ITYPE)- 1)/ ISIZE( 2)+ 3
      IF( I .GT. LMAX) GO TO 10
      ISTAK( I- 1)= ITYPE
      ISTAK( I)= LNOW
      LOUT= LOUT+ 1
      LNOW= I
      LUSED= MAX0( LUSED, LNOW)
      RETURN
C
   10 IERR= 6
      WRITE( *, 9999) I
 9999 FORMAT( 1H , 39HOVERFLOW OF COMMON ARRAY ISTAK --- NEED,
     I10)
      WRITE( *, 9998)( ISTAK( J), J= 1, 10), ISTAK( LNOW- 1),
     1 ISTAK( LNOW)
```

```
9998  FORMAT( 12I6)
      STOP
      END
C
C-----------------------------------------------------------------------
C SUBROUTINE:  ISTKRL(K)
C DE-ALLOCATES THE LAST K WORKING STORAGE AREAS
C-----------------------------------------------------------------------
C
      SUBROUTINE ISTKRL( K)
C
      COMMON/ CSTAK/ DSTAK( 2500)
C
      DOUBLE PRECISION DSTAK
      INTEGER ISTAK( 5000)
C
      EQUIVALENCE( DSTAK( 1), ISTAK( 1))
      EQUIVALENCE( ISTAK( 1), LOUT)
      EQUIVALENCE( ISTAK( 2), LNOW)
      EQUIVALENCE( ISTAK( 3), LUSED)
      EQUIVALENCE( ISTAK( 4), LMAX)
      EQUIVALENCE( ISTAK( 5), LBOOK)
C
      IN= K
C
      IF( LBOOK .LE. LNOW .AND. LNOW .LE. LUSED .AND. LUSED .LE.
     1 LMAX) GO TO 10
      IERR= 6
      WRITE( *, 9999)
9999 FORMAT( 53H WARNING...ISTAK(2),ISTAK(3),ISTAK(4) OR ISTAK(5) HIT)
      WRITE( *, 9997)( ISTAK( J), J= 1, 10), ISTAK( LNOW- 1),
     1 ISTAK( LNOW)
C
   10 IF( IN .LE. 0) RETURN
      IF( LBOOK .GT. ISTAK( LNOW) .OR. ISTAK( LNOW) .GE. LNOW- 1)
     1 GO TO 20
      LOUT= LOUT- 1
      LNOW= ISTAK( LNOW)
      IN= IN- 1
      GO TO 10
C
   20 IERR= 6
      WRITE( *, 9998)
9998 FORMAT( 45H WARNING...POINTER AT ISTAK(LNOW) OVERWRITTEN,/,
     1 11X, 27HDE-ALLOCATION NOT COMPLETED)
      WRITE( *, 9997)( ISTAK( J), J= 1, 10), ISTAK( LNOW- 1),
     1 ISTAK( LNOW)
9997  FORMAT( 12I6)
      RETURN
C
      END
```